Sun Java Enterprise System 5 Überwachungshandbuch



Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Teilenr.: 820–1587–10
März 2007

070426@17039

Inhalt

Vorwort

Dieses Handbuch beschreibt die neue Überwachungsfunktionalität von Sun Java™ Enterprise System 5 (Java ES). Die Überwachung wird vom Sun Java System Monitoring Framework 2.0 und der Sun Java System Monitoring Console 1.0 implementiert.

Die in diesem Handbuch beschriebenen Vorgänge zeigen Ihnen, wie Sie das Monitoring Framework für jede Ihrer installierten Komponenten aktivieren und anschließend alle überwachten Daten in der Monitoring Console anzeigen. Dieses Handbuch dokumentiert weder Protokolldateien noch andere Überwachungsmechanismen einzelner Komponenten außerhalb von Monitoring Framework.

An wen richtet sich dieses Handbuch

Dieses Handbuch ist bestimmt für:

- Software-Architekten, die für Java ES-Bereitstellungen Wartungspläne entwickeln müssen.
- Systemadministratoren, die Java ES installieren und konfigurieren.
- Systemadministratoren und Techniker, die Java ES-Bereitstellungen überwachen und warten.

Bevor Sie dieses Handbuch lesen

Sie sollten mit dem im nächsten Abschnitt beschriebenen Java ES-Dokumentationssatz vertraut sein. Darüber hinaus sollte Ihnen der Aufbau und die Funktionsweise der Java ES-Komponenten, die Sie überwachen wollen, vertraut sein.

Wenn Sie außerdem die Überwachungskomponenten installieren und konfigurieren wollen, muss zuerst die Installation aller anderen Komponenten abgeschlossen sein. Vor dem Ausführen einer Installation bzw. Konfiguration sollten Sie zunächst die *Sun Java Enterprise System 5 Versionshinweise für UNIX* durcharbeiten.

Java ES-Dokumentationssatz

In der Dokumentation zu Java ES wird die Bereitstellungsplanung und die Systeminstallation beschrieben. Der URL für die Systemdokumentation lautet `http://docs.sun.com/coll/1286.2`. Die in der folgenden Tabelle aufgelisteten Handbücher bieten eine Einführung in Java ES. Lesen Sie die Handbücher in der hier gezeigten Reihenfolge.

TABELLE P–1 Java Enterprise System-Dokumentation

Dokumenttitel	Inhalt
Sun Java Enterprise System 5 Versionshinweise für UNIX *Sun Java Enterprise System 5 Release Notes for Microsoft Windows*	Enthält die neuesten Informationen über Java ES, einschließlich bekannter Probleme. Darüber hinaus haben Komponenten ihre eigenen Versionshinweise, die in der Versionshinweis-Dokumentensammlung (`http://docs.sun.com/coll/1315.2`) enthalten sind.
Sun Java Enterprise System 5 – Technische Übersicht	Bietet eine Einführung in die technischen Grundlagen und in das Konzept von Java ES. Beschreibt die Komponenten, die Architektur sowie Prozesse und Funktionen.
Sun Java Enterprise System Deployment Planning Guide	Bietet eine Einführung in die Planung und die Konzeptentwicklung von Bereitstellungslösungen für Unternehmen, die auf Java ES basieren. Beschreibt die Grundlagen und Prinzipien der Bereitstellungsplanung und des Bereitstellungskonzepts sowie den Lebenszyklus einer Lösung und bietet erstklassige Beispiele und Strategien, die Sie bei der Planung der unternehmensweiten Bereitstellung von Lösungen mit Java ES anwenden können.
Sun Java Enterprise System 5 – Handbuch zur Installationsplanung	Unterstützt Sie bei der Entwicklung der Implementierungsspezifikationen für die Hardware, das Betriebssystem sowie für die Netzwerkaspekte Ihrer Java ES-Bereitstellung. Beschreibt Probleme, wie beispielsweise Komponentenabhängigkeiten, die Sie in Ihrem Installations- und Konfigurationsplan berücksichtigen müssen.
Sun Java Enterprise System 5 Installationshandbuch für UNIX *Sun Java Enterprise System 5 Installation Guide for Microsoft Windows*	Führt Sie durch die Installation von Java ES. Bietet außerdem Anweisungen zur Konfiguration der Komponenten nach der Installation und zur Überprüfung der richtigen Funktionsweise der Komponenten.
Sun Java Enterprise System 5 Installationsanweisungen für UNIX	Enthält zusätzliche Informationen zu Konfigurationsparametern, Arbeitsblätter zur Verwendung bei der Konfigurationsplanung sowie Referenzmaterial wie z. B. Standardverzeichnisse und Portnummern auf den Betriebssystemen Solaris und Linux.

TABELLE P–1 Java Enterprise System-Dokumentation *(Fortsetzung)*

Dokumenttitel	Inhalt
Sun Java Enterprise System 5 - Aufrustungshandbuch fur UNIX *Sun Java Enterprise System 5 Upgrade Guide for Microsoft Windows*	Enthält Anweisungen zur Durchführung von Upgrades auf Java ES 5 von früher installierten Versionen.
Sun Java Enterprise System 5 Überwachungshandbuch	Enthält Anweisungen zum Einrichten des Monitoring Framework für jede Produktkomponente und Verwenden der Monitoring Console zum Anzeigen von Daten in Echtzeit sowie Erstellen von Überwachungsregeln.
Sun Java Enterprise System Glossary	Definiert die in der Java ES-Dokumentation verwendeten Begriffe.

Standardpfade und Dateinamen

Die folgende Tabelle enthält die Standardpfade und -dateinamen der Java ES-Komponenten, die die Überwachung implementieren.

TABELLE P–2 Standardpfade und Dateinamen

Platzhalterkonventionen	Beschreibung	Standardwert
mfwk-Basisverzeichnis	Das Verzeichnis, in das die gemeinsam genutzte Komponente Monitoring Framework automatisch installiert wird. Dieser Pfad ist auch Teil des Konfigurationsverzeichnisses.	Solaris: `/opt/SUNWmfwk` Linux: `/opt/sun/mfwk`
MConsole-Basisverzeichnis	Das Installationsverzeichnis für die Monitoring Console.	Solaris: `/opt/SUNWjesmc` Linux: `/opt/sun/jesmc`
WebConsole-Basisverzeichnis	Das Verzeichnis, in das die gemeinsam genutzte Komponente Web Console automatisch installiert wird.	Solaris: `/etc/webconsole/console` Linux: `/etc/opt/webconsole/console`
AccessMgr-Basisverzeichnis	Das Installationsverzeichnis für Sun Java System Access Manager.	Solaris: `/opt/SUNWam` Linux: `/opt/sun/identity`
AppServer-Basisverzeichnis	Das Installationsverzeichnis für Sun Java System Anwendungsserver.	Solaris: `/opt/SUNWappserver/appserver` Linux: `/opt/sun/appserver`
CalServ-Basisverzeichnis	Das Installationsverzeichnis für Sun Java System Calendar Server.	Solaris: `/opt/SUNWics5` Linux: `/opt/sun/calendar`

TABELLE P–2 Standardpfade und Dateinamen *(Fortsetzung)*

Platzhalterkonventionen	Beschreibung	Standardwert
DirServ-Basisverzeichnis	Das Installationsverzeichnis für Sun Java System Directory Server.	Solaris: `/opt/SUNWdsee/ds6` Linux: `/opt/sun/ds6`
IM- Basisverzeichnis	Das Installationsverzeichnis für Sun Java System Instant Messaging.	Solaris: `/opt/SUNWiim` Linux: `/opt/sun/im`
MsgServ-Basisverzeichnis	Das Installationsverzeichnis für Sun Java System Messaging Server .	Solaris: `/opt/SUNWmsgsr` Linux: `/opt/sun/messaging`
Portal-Basisverzeichnis	Das Installationsverzeichnis für Sun Java System Portal Server.	Solaris: `/opt/SUNWportal` Linux: `/opt/sun/portal`
WebServer-Basisverzeichnis	Das Installationsverzeichnis für Sun Java SystemWeb Server.	Solaris: `/opt/SUNWwbsvr7` Linux: `/opt/sun/webserver`

Typografische Konventionen

In der nachfolgenden Tabelle werden die in diesem Handbuch verwendeten typografischen Änderungen erläutert.

TABELLE P–3 Typografische Konventionen

Schriftart	Bedeutung	Beispiel
`AaBbCc123`	Die Namen von Befehlen, Dateien, Verzeichnissen und Bildschirmausgaben.	Bearbeiten Sie die Datei `login`. Verwenden Sie den Befehl `ls -a`, um alle Dateien aufzulisten. `Rechnername% - Sie haben Post.`
`AaBbCc123`	Eingabe durch den Benutzer (im Gegensatz zur Computerausgabe auf dem Bildschirm	`Rechnername%` **`su`** `Passwort:`
AaBbCc123	Ein Platzhalter, der durch einen echten Namen oder Wert ersetzt werden soll.	Der Befehl zum Löschen einer Datei ist `rm` *Dateiname*.
AaBbCc123	Buchtitel, neue Begriffe und hervorzuhebende Begriffe (Beachten Sie, dass einige hervorgehobene Begriffe online fett angezeigt werden)	Lesen Sie Kapitel 6 im *Benutzerhandbuch*. Ein *cache* ist eine Kopie, die lokal gespeichert wird. Speichern Sie die Datei *nicht*.

Shell-Eingabeaufforderungen in Befehlsbeispielen

Die folgende Tabelle enthält eine Liste mit Standardsystemaufforderungen und Superuser-Eingabeaufforderungen.

TABELLE P–4 Shell-Eingabeaufforderungen

Shell	Prompt
C-Shell auf UNIX- und Linux-Systemen	`machine_name%`
C-Shell-Superuser auf UNIX- und Linux-Systemen	`machine_name#`
Bourne-Shell und Korn-Shell auf UNIX- und Linux-Systemen	`$`
Bourne-Shell- und Korn-Shell-Superuser auf UNIX- und Linux-Systemen	`#`
Microsoft Windows-Befehlszeile	`C:\`

Symbolkonventionen

In der folgenden Tabelle werden die in diesem Handbuch verwendeten Symbole erläutert.

TABELLE P–5 Symbolkonventionen

Symbol	Beschreibung	Beispiel	Bedeutung
`[ ]`	Enthält optionale Argumente und Befehlsoptionen.	`ls [-l]`	Die Option `-l` ist nicht erforderlich.
`{ | }`	Enthält verschiedene Möglichkeiten für eine erforderliche Befehlsoption.	`-d {y|n}`	Für die Option `-d` muss entweder das Argument `y` oder das Argument `n` verwendet werden.
`${ }`	Gibt einen Variablenverweis an.	`${com.sun.javaRoot}`	Verweist auf den Wert der Variablen `com.sun.javaRoot`.
-	Gibt eine Kombination aus mehreren Tasten an.	STRG-A	Halten Sie die Taste STRG gedrückt, während Sie die Taste A drücken.
+	Gibt die aufeinander folgende Betätigung mehrerer Tasten an.	STRG+A+N	Drücken Sie die Taste STRG, lassen Sie sie los und drücken Sie anschließend die nachfolgenden Tasten.

TABELLE P–5 Symbolkonventionen (Fortsetzung)

Symbol	Beschreibung	Beispiel	Bedeutung
→	Zeigt die Auswahl eines Menüelements in einer grafischen Benutzeroberfläche an.	Datei → Neu → Vorlagen	Wählen Sie im Menü "Datei" die Option "Neu" aus. Wählen Sie im Untermenü "Neu" die Option "Vorlagen" aus.

Dokumentation, Support und Schulung

Auf der Sun-Website finden Sie Informationen zu den folgenden zusätzlichen Ressourcen:

- Dokumentation (`http://www.sun.com/documentation/`)
- Support (`http://www.sun.com/support/`)
- Schulung (`http://www.sun.com/training/`)

Durchsuchen von Sun-Produktdokumentation

Neben dem Durchsuchen der Sun-Produktokumentation unter docs.sun.com^SM können Sie in Suchmaschinen mithilfe der folgenden Syntax suchen:

Suchbegriff `site:docs.sun.com`

Wenn Sie beispielsweise nach dem Begriff broker suchen möchten, sollten Sie eingeben:

```
broker site:docs.sun.com
```

Wenn auch andere Sun-Websites durchsucht werden sollen (z. B. java.sun.com, www.sun.com und developers.sun.com), geben Sie statt `docs.sun.com` im Suchfeld `sun.com` ein.

Verweise auf Drittanbieter-Websites

In der vorliegenden Dokumentation wird auf URLs von Drittanbietern verwiesen, über die zusätzliche relevante Informationen zur Verfügung gestellt werden.

Hinweis – Sun ist für die Verfügbarkeit der in diesem Dokument erwähnten Drittanbieter-Websites nicht verantwortlich. Sun unterstützt keine Inhalte, Werbung, Produkte oder sonstige Materialien, die auf oder über solche Websites oder Ressourcen verfügbar sind, und übernimmt keine Verantwortung oder Haftung dafür. Sun ist nicht verantwortlich oder haftbar für tatsächliche oder vermeintliche Schäden oder Verluste, die durch oder in Verbindung mit der Verwendung von über solche Websites oder Ressourcen verfügbaren Inhalten, Waren oder Dienstleistungen bzw. dem Vertrauen darauf entstanden sind oder angeblich entstanden sind.

Kommentare sind willkommen

Sun möchte seine Dokumentation laufend verbessern. Ihre Kommentare und Vorschläge sind daher immer willkommen. Wenn Sie Kommentare abgeben möchten, rufen Sie die Seite `http://docs.sun.com` und klicken Sie auf "Kommentare senden". Geben Sie im Online-Formular den Dokumenttitel und die Teilenummer an. Die Teilenummer ist eine sieben- oder neunstellige Zahl, die Sie auf der Titelseite des Buchs oder im Dokument-URL finden. Die Teilenummer dieses Buchs lautet z. B. 820-1587-10 .



KAPITEL 1

Überblick über die Überwachung mit Java ES

In diesem Handbuch werden die Komponenten Monitoring Framework 2.0 und Monitoring Console 1.0 des Sun Java™ Enterprise System (Java ES) beschrieben. Diese Komponenten implementieren zusammen das neue, in Version 5 eingeführte Leistungsmerkmal zur Überwachung.

Die in diesem Handbuch enthaltenen Anleitungen zeigen Ihnen, wie Sie das Monitoring Framework für jede Ihrer installierten Komponenten aktivieren und anschließend alle überwachten Daten in der Monitoring Console anzeigen. Dieses Handbuch dokumentiert keine Protokolldateien, Fehlermeldungen oder andere Überwachungsmechanismen, die einige Komponenten womöglich außerhalb dieses Frameworks implementieren. Das Monitoring Framework und die Monitoring Console bieten keine Verwaltungsfunktionen für überwachte Komponenten. Informationen zur Verwaltung von Komponenten finden Sie in der Dokumentation des jeweiligen Produkts.

In diesem Kapitel werden das Prinzip der Überwachung sowie der Aufbau des Monitoring Framework vorgestellt.

Dieses Kapitel enthält die folgenden Abschnitte:

- „Die Komponenten des Monitoring Framework und der Monitoring Console " auf Seite 15
- „Funktionsprinzip der Java ES-Überwachung" auf Seite 16
- „Empfohlene Installationsreihenfolge" auf Seite 20

Die Komponenten des Monitoring Framework und der Monitoring Console

Das Sun Java System Monitoring Framework stellt die Infrastruktur zur Instrumentierung von Komponenten und zum Verfügbarmachen von Attributen für die Überwachung bereit. Es definiert eine Hierarchie überwachter Objekte namens *Common Monitoring Model (CMM)*, die auf der Industritstandard-Spezifikation "Common Information Model" (CIM) beruht. Jede

Produktkomponente macht die Objekte mit überwachbaren Attributen zugänglich, und ein sog. *Knotenagent* stellt eine Ansicht mehrerer Komponenten auf einem Host zusammen. Das Monitoring Framework bietet darüber hinaus einen Mechanismus zum Erfassen von Betriebsstatistiken und Definieren von Alarmen gemäß benutzerspezifischer Schwellenwerte.

Die Sun Java System Monitoring Console ist die grafische Benutzeroberfläche zum Überwachen von Java ES-Komponenten. Sie umfasst einen *Master-Agent*, der mit allen in einer Java ES-Bereitstellung enthaltenen Knotenagenten kommuniziert. Die Monitoring Console ist eine Internet-basierte Anwendung, auf die über die Sun Java SystemWeb Console mithilfe des HTTP-Protokolls von überallher zugegriffen werden kann. Im Hauptbildschirm wird eine Übersicht über alle aktivierten Komponenten einschließlich aller ausgelösten Alarme angezeigt. Sie können dann auf die Hierarchie überwachter Objekte in jeder Komponente zugreifen und den ausführlichen Status sowie die Echtzeitwerte aller überwachten Attribute anzeigen. Die Monitoring Console-Oberfläche ermöglicht das Anzeigen ausführlicher Informationen zu jedem Alarm, das Quittieren von Alarmen bzw. das Erstellen neuer Überwachungsregeln auf der Grundlage beliebiger Attribute.

Funktionsprinzip der Java ES-Überwachung

Unter Überwachung versteht man den Prozess des Erfassens von Laufzeitdaten, Zugänglichmachens dieser Daten und anschließendem Berechnen von Servicekriterien, sodass Systemadministratoren die Leistung bewerten und im Alarmfall entsprechend benachrichtigt werden können. Während des Betriebs müssen Administratoren nur mit der Monitoring Console interagieren, um Leistungsstatistiken anzuzeigen, Regeln zur automatischen Überwachung zu erstellen und Alarme zu quittieren. Für die Konfiguration, Fehlerbehebung und fortgeschrittene Überwachung ist das Verständnis des Aufbaus des Monitoring Framework und seiner Kommunikation mit der Monitoring Console jedoch von großem Vorteil.

Die Überwachung in Java ES beruht auf den folgenden Prinzipien:

- Das Common Monitoring Model (CMM) gewährleistet, dass alle Java ES-Komponenten für die Überwachung gleichartige Objekte und Werte vergleichbarer Attribute zur Verfügung stellen.
- Von den CMM-Schnittstellen definierte Java-Objekte bieten standardisierte Instrumentierungen für Produktkomponenten.
- Knotenagenten machen alle überwachten Objekte für alle auf einem System installierten Komponenten zugänglich und verwalten die Statistik, Regeln und Alarme für diese Objekte.
- Ein auf einem eigenen Host-Rechner installierter Master-Agent sammelt die Daten aller überwachten Objekte von den Knotenagenten und leitet diese Daten an die Monitoring Console weiter.

In den folgenden Abschnitten werden die einzelnen Konzepte der Überwachungsarchitektur ausführlicher erläutert.

Das Common Monitoring Model (CMM)

Die Grundlage eines standardisierten Überwachungsmechanismus bildet die Definition, welche Objekte überwacht werden sollen und die Übernahme dieser Objekte für alle überwachten Komponenten. Zu diesem Zweck definiert die Überwachungsarchitektur das Common Monitoring Model (CMM) als Erweiterung des Common Information Model (CIM), das von der Distributed Management Task Force (DMTF) unterhalten wird. CMM ist ein Informationsmodell, das überwachte Objekte wie z. B. Computer, Anwendungen usw. sowie ein Datenmodell für gleichartige Werte wie z. B. Betriebsstatuswerte definiert. Im Rahmen des Informationsmodells definiert CMM darüber hinaus die Attribute eines Objekts, z. B. die von einem Dienst zu bearbeitenden Anforderungen, sowie die Beziehungen zwischen Objekten wie z. B. die Tatsache, dass ein Dienst auf einem bestimmten Computer installiert ist.

Dank des CMM sind Konzepte wie z. B: Anwendungen, Dienste, Zugriffspunkte usw. für alle Produktkomponenten gleich, auch wenn die zugrunde liegende Implementierung unterschiedlich ist. So kann Web Server beispielsweise einen Dienst zur Behandlung von HTTP-Anforderungen zugänglich machen, während Directory Server einen Dienst zur Behandlung von LDAP-Anforderungen bietet. Ein diesem Standard entsprechendes Objekt erfasst jedoch nur das Gemeinsame dieser Funktionen, so z. B. die Anzahl der behandelten Anforderungen oder die Zeit, die innerhalb eines bestimmten Zeitraums im Durchschnitt zur Behandlung einer Anforderung notwendig ist usw.

Darüber hinaus sind einige Datenwerte standardisiert, sodass ihre Bedeutung im gesamten System gleichartig ist. Der Betriebssstatus `DEGRADED` bedeutet beispielsweise stets, dass ein Dienst zwar noch verfügbar, seine Leistung jedoch beträchtlich abgefallen ist. Dabei spielt es keine Rolle, welche Produktkomponente überwacht wird.

Die CMM-Spezifikation ist in die für die Instrumentierung verwendeten Java-Schnittstellen und Klassen integriert. Diese sind in Anhang A, beschrieben.

CMM-Instrumentierung

Im Monitoring Framework, wird die Instrumentierung von bestimmten Java-Schnittstellen und -Klassen bereitgestellt, die die CMM-Definitionen implementieren. Für die neue Überwachungsfunktionalität von Java ES wurde der Code von Produktkomponenten entsprechend instrumentiert, damit Instanzen von CMM-Objekten erstellt und Laufzeitwerte über die Attribute überwachter Objekte zugänglich gemacht werden können . Die von jeder Komponente implementierten CMM-Objekte legen fest, was überwacht werden kann. Deswegen ist die Anzahl der für die Überwachung zugänglichen Attribute je nach Komponente unterschiedlich. Die Liste der Objekte und Attribute, die von jeder Produktkomponente zugänglich gemacht werden, finden Sie in Anhang B.

Knotenagenten

In der Begriffswelt der Überwachung versteht man unter einem Knoten einen einzelnen logischen Host, der durch einen vollständigen Domänennamen bzw. eine IP-Adresse eindeutig identifiziert werden kann. Ein Knoten kann entweder ein gesamtes System oder eine als virtuelles System konfigurierte Solaris Zone sein. Der Knotenagent kommuniziert mit allen instrumentierten Komponenten auf diesem Host und macht alle ihre überwachten Objekte zugänglich. Darüber hinaus verwalten Knotenagenten die gesamte Logik zum Erfassen von Leistungsdaten, überwachen in Regeln definierte Schwellenwerte und generieren Alarme für die in ihnen enthaltenen überwachten Objekte.

Das folgende Schema veranschaulicht den Inhalt eines Knotenagenten auf einem einzelnen Host mit Instanzen von drei Java ES-Produktkomponenten. Es zeigt außerdem, wie die Instrumentierung im Knotenagenten instanziiert wird, um die von den Produktkomponenten bereitgestellten Werte für die Überwachung zugänglich zu machen.

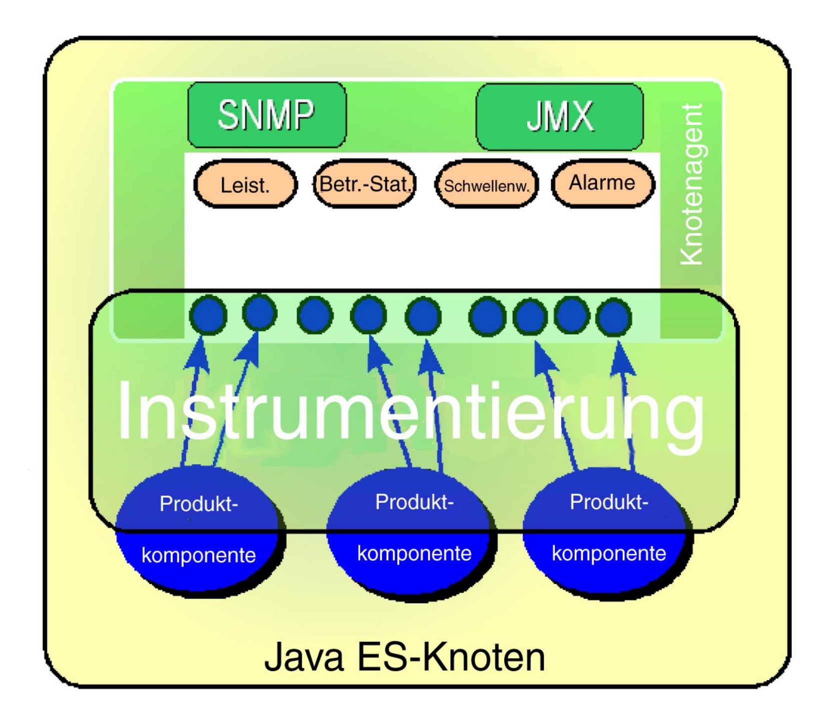


ABBILDUNG 1–1 Schematische Darstellung eines Knotenagenten

Ein Knotenagent ist als Modul implementiert, das in den Common Agent Container (der eine Java Virtual Machine darstellt) geladen wird . Die Implementierung von Knotenagenten beruht auf den Java Management Extensions (JMX), des Java-Standarderweiterungspakets für die Überwachung und Fernverwaltung. Alle JMX-fähigen und CMM-kompatiblen Überwachungsanwendungen können auf die überwachten Objekte im Knotenagent zugreifen.

Mithilfe der JMX-Funktionalität kann der Knotenagent darüber hinaus über das Simple Network Monitoring Protocol (SNMP) bestimmte überwachte Objekte zugänglich machen.

Der Master-Agent

Der Master-Agent wird als Teil der Monitoring Console-Installation auf einem eigenen Rechner bereitgestellt. Der Master-Agent wird mit dem Namen und der Adresse aller Knoten konfiguriert, sodass die überwachten Objekte von allen Knotenagenten zusammengefasst werden können. Der Master-Agent ist ebenfalls JMX-basiert und nutzt JMX zur Kommunikation mit den Knotenagenten. Er befindet sich in seinem lokalen Common Agent Container.

Die folgende Abbildung zeigt einen mit zwei Knoten verbundenen Master-Agenten. Die Monitoring Console stellt zum Überwachen der drei Komponenten auf jedem Knoten zum Master-Agent eine Verbindung her. Wenn Sie zur Überwachung SNMP verwenden möchten, müssen Sie zu jedem einzelnen Knoten getrennt eine Verbindung herstellen, da der Master-Agent keine SNMP-Attribute zusammenfasst. Der Master-Agent ist nur zur Verwendung mit der Monitoring Console gedacht und für andere Überwachungsanwendungen nicht zugänglich.

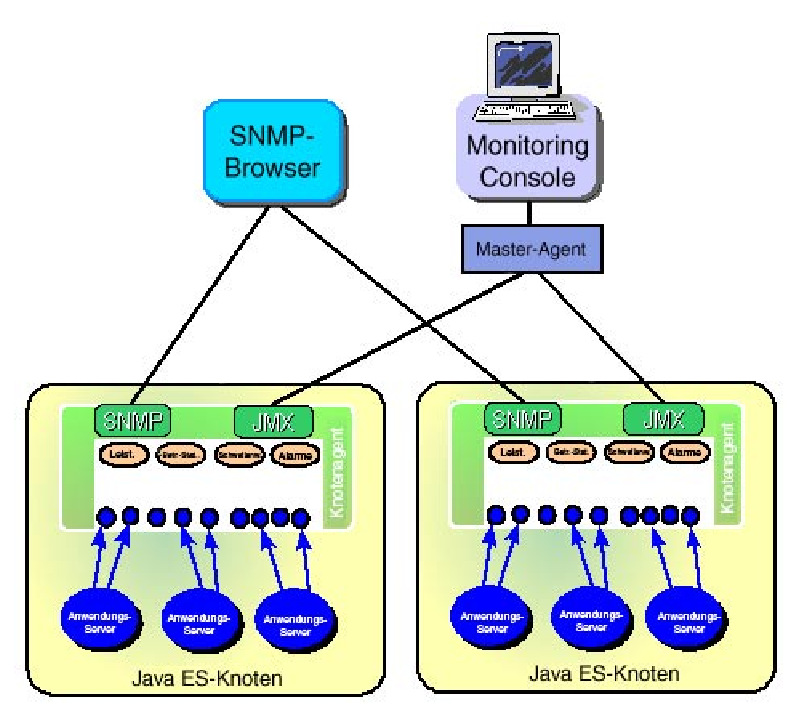


ABBILDUNG 1–2 Schematische Darstellung der gesamten Überwachungsarchitektur

Empfohlene Installationsreihenfolge

Wenn Sie die Java ES-Überwachungsfunktionalität bewerten oder bereitstellen wollen, wird empfohlen, die Installation in der folgenden Reihenfolge durchzuführen:

1. Installation und Konfiguration aller Komponenten in Ihrer Bereitstellung gemäß den Empfehlungen und Anweisungen im *Sun Java Enterprise System 5 Installationshandbuch für UNIX*.
2. Aktivieren und Konfigurieren des Monitoring Framework für alle überwachten Komponenten (siehe Kapitel 2).
3. Installation der Monitoring Console auf einem eigenen Host, Starten des Master-Agenten, und anschließend des Web-Servers (siehe Kapitel 3. Alle überwachten Komponenten sollten in der Monitoring Console danach sichtbar sein und aktiv überwacht werden.

Hinweis – Wegen einer Inkompatibilität von Knotenagent und Master-Agent in in dieser Version muss die Monitoring Console auf einem Host installiert werden, auf dem keine anderen Java ES-Komponenten installiert sind. Ausführlichere Informationen finden Sie im Abschnitt „Fehlerbehebung an der Monitoring Console" auf Seite 70.

Immer, wenn Sie nach dem Aktivieren der Überwachung an bereitgestellten Komponenten Änderungen vorgenommen haben, müssen Sie den Container für den Master-Agenten und den Web-Server für die Monitoring Console neu starten (siehe „Fehlerbehebung am Monitoring Framework" auf Seite 30).



KAPITEL 2

Aktivieren und Konfigurieren des Monitoring Framework

Das Monitoring Framework stellt für jede einzelne überwachte Komponente die Instrumentierung und den Knotenagenten bereit (siehe Abschnitt „Funktionsprinzip der Java ES-Überwachung” auf Seite 16). Deswegen handelt es sich beim Monitoring Framework um eine gemeinsam genutzte Komponente, die bei jeder Installation einer überwachten Komponente mithilfe des Java Enterprise System-Installationsprogramms automatisch installiert wird.

Bei vielen überwachten Komponenten ist die Überwachungsfunktion jedoch standardmäßig nicht aktiviert, und einige Komponenten müssen erst noch speziell konfiguriert werden, bevor sie im Knotenagenten angezeigt werden. Für jede der Produktkomponenten, die Sie installiert haben, müssen Sie die in diesem Kapitel enthaltenen Anweisungen abarbeiten.

Hinweis – Bevor Sie die in diesem Kapitel aufgeführten Anweisungen abarbeiten, sollten Sie zunächst die Produktkomponenten instalieren, die Sie auf einem Host ausführen möchten. Vor dem Ausführen einer Installation bzw. Konfiguration sollten Sie zunächst die *Sun Java Enterprise System 5 Versionshinweise für UNIX* durcharbeiten.

Die hier beschriebenen Vorgänge nutzen den Befehl `mfwksetup`, der normalerweise nicht gebraucht wird und deswegen nicht dokumentiert ist.

Dieses Kapitel enthält die folgenden Abschnitte:

- „Einrichten des Common Agent Container" auf Seite 28
- „Fehlerbehebung am Monitoring Framework" auf Seite 30
- „Der Befehl `mfwkadm`" auf Seite 31

Struktur der installierten Verzeichnisse

Als gemeinsam genutzte Komponente wird das Monitoring Framework automatisch immer dann installiert, wenn es gebraucht wird. Die Bezeichnung des auf Ihrem Betriebssystem installierten Pakets entnehmen Sie bitte aus Kapitel 5, „Liste der installierbaren Pakete" in *Sun Java Enterprise System 5 Installationsanweisungen für UNIX*. In der folgenden Tabelle sind die im Monitoring Framework-Paket enthaltenen Verzeichnisse beschrieben. Der Parameter für das Standard-Installationsverzeichnis *mfwk-Basisverzeichnis* steht für die folgenden Verzeichnisse (siehe „Standardpfade und Dateinamen" auf Seite 9):

- Solaris: `/opt/SUNWmfwk`
- Linux: `/opt/sun/mfwk`

TABELLE 2–1 Vom Monitoring Framework verwendete Verzeichnisse

Pfad	Beschreibung
mfwk-Basisverzeichnis`/config`	Vorlage für eine Konfigurationsdatei
Solaris: *mfwk-Basisverzeichnis*`/lib` Linux: *mfwk-Basisverzeichnis*`/share/lib`	Java-Archivdateien (`.jar`)
Solaris: *mfwk-Basisverzeichnis*`/lib` Linux: *mfwk-Basisverzeichnis*`/share/lib`	32–Bit-Laufzeitbibliotheken (`.so`)
Solaris (SPARC®-Systeme): *mfwk-Basisverzeichnis*`/lib/sparcv9` Solaris (x86-Systeme): *mfwk-Basisverzeichnis*`/amd64` Linux: *mfwk-Basisverzeichnis*`/lib64`	64–Bit-Laufzeitbibliotheken (`.so`)
mfwk-Basisverzeichnis`/bin`	Öffentliche Skripten und private Binärdateien
mfwk-Basisverzeichnis`/mib`	Textversionen von SNMP MIBs, die vom Monitoring Framework unterstützt werden
mfwk-Basisverzeichnis`/xml`	Common Agent Container-Deskriptorvorlagen für Knoten- und Master-Agent (bereitgestellt über den Befehl `mfwksetup`)
mfwk-Basisverzeichnis`/dtd`	DTD-Dateien für die OSS/J-Funktionalität.

TABELLE 2–1 Vom Monitoring Framework verwendete Verzeichnisse *(Fortsetzung)*

Pfad	Beschreibung
`/etc/`*mfwk-Basisverzeichnis*`/config`	Konfigurationsdateien, u. a. auch für Sicherheitsfunktionen
`/etc/`*mfwk-Basisverzeichnis*`/xml`	Common Agent Container-Deskriptoren für Agenten und Beispiele
`/var/`*mfwk-Basisverzeichnis*`/logs`	Protokolldateien des Monitoring Framework
`/var/`*mfwk-Basisverzeichnis*`/reports`	Basisverzeichnis für Überwachungsregelberichte
`/var/`*mfwk-Basisverzeichnis*`/alarms`	Speicherort für Alarmdateien

Verwendung des Monitoring Framework mit Access Manager

In Access Manager ist die Überwachung zwar standardmäßig aktiviert, aufgrund einer Einschränkung werden die überwachten Objekte aber nicht in der Monitoring Console angezeigt.

Eine Liste mit Objekten und Attributen, die überwacht werden können, finden Sie im Abschnitt „Instrumentierung von Access Manager" auf Seite 77.

▼ Aktivieren der Überwachung in Access Manager

1 Deaktivieren Sie mit den folgenden Befehlen die Überwachung in Access Manager vorübergehend:

```
cacaoadm unregister-module com.sun.cmm.am.xml
cacaoadm restart
```

2 Öffnen Sie die XML-Deskriptordatei von Access Manager zum Bearbeiten:

```
vi /etc/AccessMgr-Basisverzeichnis/config/com.sun.cmm.am.xml
```

3 Suchen Sie die Zeilen, die Folgendes enthalten:

```
<param-name>Product Name</param-name>
<param-value>Access Manager</param-value>
```

Ändern Sie die zweite Zeile wie folgt:

```
<param-value>Java ES Access Manager</param-value>
```

Speichern Sie die Datei und schließen Sie den Editor.

4 Registrieren Sie das geänderte XML-Modul:

```
mfwk-Basisverzeichnis/bin/mfwksetup -u /etc/AccessMgr-Basisverzeichnis/config/com.sun.cmm.am.xml
mfwk-Basisverzeichnis/bin/mfwksetup -r /etc/AccessMgr-Basisverzeichnis/config/com.sun.cmm.am.xml
```

5 Starten Sie den Common Agent Container neu:

```
cacaoadm restart
```

Allgemeine Fehler Aufgrund des ungetesteten Verhaltens mit Web-Containern von Drittanbietern ist die Überwachung standardmäßig deaktiviert, wenn Access Manager in Websphere bzw. Weblogic bereitgestellt wird. Sie können die Überwachung zwar gemäß der im Abschnitt „Selektives Deaktivieren und Neuaktivieren der Überwachung" auf Seite 61 beschriebenen Anleitung aktivieren, diese Konfiguration wird jedoch nicht unterstützt.

Verwendung des Monitoring Framework mit Anwendungsserver

Eine Liste mit Objekten und Attributen, die überwacht werden können, finden Sie im Abschnitt „Instrumentierung von Anwendungsserver" auf Seite 77

▼ Aktivieren der Überwachung in Anwendungsserver

1 Bearbeiten Sie die Datei `/var/`*AppServer-Basisverzeichnis*`/domains/domain1/config/domain.xml` **und ändern Sie alle Einstellungen in** `module-monitoring-level` **von** `OFF` **auf** `HIGH`**. Alternativmethode:**

a. Melden Sie sich an der Administrationskonsole von Anwendungsserver unter `https://`*hostname*`:4849` **an.**

b. Klicken Sie auf "Configurations", und dann auf "server-config (Admin Config)".

c. Setzen Sie den Überwachungswert auf `HIGH`**.**

d. Setzen Sie alle anderen Werte auf `HIGH`**.**

2 Starten Sie Anwendungsserver mit den folgenden Befehlen neu:

```
cd AppServer-Basisverzeichnis/appserv/bin
asadmin stop-domain domain1
asadmin start-domain user meinBenutzername domain1
```

Geben Sie das Passwort für *meinBenutzername* ein, wenn Sie dazu aufgefordert werden.

3 **Wenn Sie mit Anwendungsserver eine Instanz von Portal Server bereitgestellt und überwacht haben, stört der Neustart von Anwendungsserver die Überwachung von Portal Server. Damit die Portal Server-Instanz in der Monitoring Console sichtbar wird, müssen Sie in einem Browser eine Portalseite aufrufen. Öffnen Sie beispielsweise die Seite** `http://portalserv.example.com:8080/portal` **, um die Überwachung von** `portalserv.example.com` **zu ermöglichen.**

Allgemeine Fehler Aufgrund einer Einschränkung werden die überwachten Objekte für Anwendungsserver aus der Monitoring Framework entfernt, wenn der Anwendungsserver abstürzt oder nicht läuft. Wenn dies der Fall ist, verschwindet der Anwendungsserver aus der Monitoring Console und kann nicht mehr überwacht werden.

Verwendung des Monitoring Framework mit Calendar Server

Eine Liste mit Objekten und Attributen, die überwacht werden können, finden Sie im Abschnitt „Instrumentierung von Calendar Server" auf Seite 77.

▼ Aktivieren der Überwachung in Calendar Server

1 **Bearbeiten Sie die Datei** `ics.conf`**.**

vi *CalServ-Basisverzeichnis*/cal/config/ics.conf

2 **Fügen Sie die folgende Zeile hinzu:**

```
local.mfagent.enable="yes"
```

3 **Registrieren Sie das XML-Modul von Calendar Server:**

mfwk-Basisverzeichnis/bin/mfwksetup -r /opt/SUNWics5/cal/lib/com.sun.cmm.cs.xml

4 **Setzen Sie die Umgebungsvariable** `LD_LIBRARY_PATH` **wie folgt:**

LD_LIBRARY_PATH=*mfwk-Basisverzeichnis*/lib:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH

5 **Starten Sie Calendar Server neu:**

cd *CalServ-Basisverzeichnis*/cal/sbin/
./stop-cal
./start-cal

6 **Starten Sie den Common Agent Container neu:**

```
cacaoadm restart
```

Verwendung des Monitoring Framework mit Directory Server

Eine Liste mit Objekten und Attributen, die überwacht werden können, finden Sie im Abschnitt „Instrumentierung von Directory Server" auf Seite 78.

▼ Aktivieren der Überwachung in Directory Server

1. **Erstellen Sie eine temporäre Passwortdatei:**

   ```
   echo –n Passwort > /tmp/pwd
   ```

2. **Aktivieren Sie das Überwachungs-Plugin mit dem folgenden Befehl:**

   ```
   DirServ-Basisverzeichnis/ds6/bin/dscfg enable-plugin -e -p 389 -w /tmp/pwd  "Monitoring Plugin"
   ```

3. **Starten Sie Directory Server neu:**

   ```
   cd DirServ-Basisverzeichnis/ds6/bin
   ./dsadm restart /var/DirServ-Basisverzeichnis/DSinstance/
   ```

Verwendung des Monitoring Framework mit Instant Messaging

Eine Liste mit Objekten und Attributen, die überwacht werden können, finden Sie im Abschnitt „Instrumentierung von Instant Messaging" auf Seite 78.

▼ Aktivieren der Überwachung mit Instant Messaging

1. **Öffnen Sie die XML-Deskriptordatei von Instant Messaging zum Bearbeiten:**

   ```
   vi /etc/IM-Basisverzeichnis/default/com.sun.cmm.im.xml
   ```

2. **Ändern Sie das Installationsverzeishnis von** *IM-Basisverzeichnis* **in** `/etc/`*IM-Basisverzeichnis*`/default`**.**

3. **Registrieren Sie den geänderten XML-Deskriptor von Instant Messaging:**

   ```
   mfwk-Basisverzeichnis/bin/mfwksetup -r /etc/IM-Basisverzeichnis/default/com.sun.cmm.im.xml
   ```

4. **Aktivieren Sie die Instrumentierung durch Hinzufügen der folgenden Zeile zur Datei** *IM-Basisverzeichnis*`/config/iim.conf`**:**

   ```
   iim_server.monitor.enable = true
   ```

5. **Starten Sie Instant Messaging mit den folgenden Befehlen neu:**

```
cd IM-Basisverzeichnis/sbin
./imadmin stop
./imadmin start
```

6. **Starten Sie den Common Agent Container neu:**

```
cacaoadm restart
```

Verwendung des Monitoring Framework mit Messaging Server

Eine Liste mit Objekten und Attributen, die überwacht werden können, finden Sie im Abschnitt „Instrumentierung von Messaging Server " auf Seite 78.

▼ Aktivieren der Überwachung in Messaging Server

1. **Aktivieren Sie die Instrumentierung mit dem folgenden Befehl:**

```
MsgServ-Basisverzeichnis/sbin/configutil -o local.mfagent.enable -v 1
```

2. **Registrieren Sie das XML-Modul von Messaging Server :**

```
mfwk-Basisverzeichnis/bin/mfwksetup -r MsgServ-Basisverzeichnis/lib/com.sun.cmm.ms.xml
```

3. **Starten Sie Messaging Server neu:**

```
cd MsgServ-Basisverzeichnis/sbin
./stop-msg
./start-msg
```

4. **Starten Sie den Common Agent Container neu:**

```
cacaoadm restart
```

Verwendung des Monitoring Framework mit Portal Server

Eine Liste mit Objekten und Attributen, die überwacht werden können, finden Sie im Abschnitt „Instrumentierung von Portal Server" auf Seite 78.

▼ Aktivieren der Überwachung in Portal Server

- **Zur Aktivierung von Portal Server müssen Sie sich unter der folgenden URL anmelden:**

  http://*Vollständiger_Hostname*:8080/portal/dt

  Dadurch werden die Portal-JSPs kompiliert, die die Portal-Instanz erstellen, die anschließend überwacht werden kann

Allgemeine Fehler Immer wenn der Anwendungsserver, auf dem der Portal Server installiert ist, neu gestartet wird, müssen Sie die Überwachung mit diesem Vorgang manuell neu starten.

Verwendung des Monitoring Framework mit Web Server

Eine Liste mit Objekten und Attributen, die überwacht werden können, finden Sie im Abschnitt „Instrumentierung von Web Server" auf Seite 78.

▼ Aktivieren der Überwachung in Web Server

1. **Starten Sie Web Server mit den folgenden Befehlen neu:**

   cd /var/*WebServer-Basisverzeichnis*/https-*Vollständiger_Hostname*/bin
   ./startserv

2. **Starten Sie den Administrationsserver:**

   cd /var/*WebServer-Basisverzeichnis*/admin-server/bin
   ./startserv

Einrichten des Common Agent Container

Der Common Agent Container ist eine weitere gemeinsam genutzte Komponente, die das Monitoring Framework zum Ausführen des Knotenagenten verwendet. Je nach der Installationsreihenfolge kann es sein, dass der Common Agent Container gestoppt und anschließend neu gestartet werden muss. Darüber hinaus wurde der Common Agent Container instrumentiert und kann ebenfalls überwacht werden. Eine Beschreibung der überwachten Objekte finden Sie im Abschnitt „Instrumentierung des Common Agent Container" auf Seite 77.

Führen Sie den folgenden Befehl aus, um zu überprüfen, ob der Common Agent Container (und folglich auch der Knotenagent) bereits gestartet wurden:

```
cacaoadm status
```

Wenn eine Meldung angezeigt wird, die der folgenden ähnelt, läuft der Knotenagent:

```
default instance is DISABLED at system startup.
Smf monitoring process:
26996
Uptime: 0 day(s), 0:57
```

Wenn eine Meldung angezeigt wird, die der folgenden ähnelt, läuft der Knotenagent nicht:

```
default instance is DISABLED at system startup.
default instance is not running.
```

▼ Aktivieren der Überwachung des Common Agent Container

Der Common Agent Container ist eine gemeinsam genutzte Komponente mit Instrumentierung für die Überwachung. Alle Java ES-Komponenten auf einem Host oder in einer Zone besitzen den gleichen Common Agent Container und Knotenagenten (siehe „Knotenagenten" auf Seite 18). Den folgenden Vorgang müssen Sie als Root-Benutzer auf jedem logischen Host in Ihrer Bereitstellung ausführen, in der der Common Agent Container überwacht werden soll.

1 Wenn der Common Agent Container läuft, müssen Sie ihn mithilfe des folgenden Befehls stoppen:

```
cacaoadm stop
```

2 Aktivieren Sie die Instrumentierung am Container selbst:

```
cacaoadm set-param enable-instrumentation=true
```

3 Überprüfen Sie den Wert des gerade eingestellten Parameters und starten Sie den Common Agent Container neu:

```
cacaoadm get-param enable-instrumentation
cacaoadm start
```

4 Erstellen Sie ein Schlüsselpasswort:

`echo –n` *Passwort* `> /etc/`*mfwk-Basisverzeichnis*`/config/security/password.cacao`

5 Generieren Sie Ihren Schlüssel:

mfwk-Basisverzeichnis`/bin/cpgenkey -n cacao -p /etc/`*mfwk-Basisverzeichnis*`/config/security/password.cacao`

6 Registrieren Sie die eigenen Überwachungsmodule des Common Agent Container:

```
cacaoadm register-module /usr/lib/cacao/ext/instrum/config/com.sun.cacao.instrum.xml
cacaoadm register-module /usr/lib/cacao/ext/instrum_jesmf/config/com.sun.cacao.instrum.jesmf.xml
```

```
cacaoadm register-module /usr/lib/cacao/ext/instrum_jesmf/config/com.sun.cacao.cmm.xml
```

Fehlerbehebung am Monitoring Framework

Bekannte Probleme sind auch in den *Sun Java Enterprise System 5 Versionshinweise für UNIX* aufgeführt.

Verwendung des Monitoring Framework auf HP-UX-Plattformen

Die Java Virtual Machine (JVM) auf HP-UX ist standardmäßig nicht für die intensive Rechenleistung, die das Monitoring Framework benötigt, ausgelegt, was zu `OutOfMemory` -Ausnahmefehlern führen kann. Zur Anpassung der JVM müssen Sie das Dienstprogramm `HPjconfig` von der folgenden URL herunterladen:`http://h21007.ww2.hp.com/dspp/tech/tech_TechDocumentDetailPage_IDX/1,1701,1620,00.html`.

Verwendung des Monitoring Framework unter Microsoft Windows

Die Überwachung von Java ES-Komponenten auf der Windows-Plattform mit dem Monitoring Framework wird vollständig unterstützt, obwohl es einige Unterschiede gibt. So müssen Sie beispielsweise auf Java 1.5 oder höher aktualisieren, um bestimmte bekannte Probleme zu vermeiden. Weitere bekannte Probleme sind in den *Sun Java Enterprise System 5 Release Notes for Microsoft Windows* aufgeführt.

▼ Neustart eines Knotenagenten

Wenn Sie den Common Agent Container, in dem ein Knotenagent installiert ist, neu starten müssen, werden die von diesem Knotenagenten überwachten Komponenten erst dann in der Monitoring Console angezeigt, wenn Sie folgenden Vorgang ausführen:

1. **Starten Sie den Common Agent Container neu, in dem der *Knotenagent* installiert ist.:**

   ```
   cacaoadm restart
   ```

2. **Starten Sie den Common Agent Container neu, in dem der *Master-Agent* installiert ist.: Der Master-Agent läuft im Monitoring Framework auf dem Host oder in der Zone, in der die Monitoring Console installiert ist.**

   ```
   cacaoadm restart
   ```

Der Master-Agent stellt automatisch die Verbindung zu allen Knotenagenten her, die er vorher überwacht hat.

3 **Starten Sie den Web-Server neu, auf dem die Monitoring Console installiert ist:**

```
/usr/sbin/smcwebserver restart
```

Der Befehl `mfwkadm`

Dieser Abschnitt enthält die deutsche Übersetzung der Man Page für den Befehl `mfwkadm`, einem Dienstprogrammbefehl in Man Page-Abschnitt 1M. Mit diesem Befehl verwalten Sie den Inhalt von Knotenagenten einschließlich der Module für überwachte Komponenten und den für die betreffenden Knoten definierten Überwachungsregeln (auch als Überwachungsaufträge bezeichnet). Einige der Begriffe und Beschreibungen in der Man Page wurden zur Anpassung an dieses Dokument geringfügig geändert.

Kurzbeschreibung

mfwkadm --help

mfwkadm start

mfwkadm stop

mfwkadm restart

mfwkadm list-params

mfwkadm list-modules

mfwkadm info *runningInstance*

Leistungsüberwachung

mfwkadm pm-job observable-classes

mfwkadm pm-job observable-objects [class=*Objektklasse*] [domain= *Objektdomäne*]

mfwkadm pm-job observable-attributes class=*Objektklasse*

mfwkadm pm-job list

mfwkadm pm-job info *Auftragsname*

mfwkadm pm-job create *Auftragsname*
granularity=*Ganzzahliger_Wert* object=*Objektname* [object=*Objektname* ...]

mfwkadm pm-job delete *Auftragsname*

mfwkadm pm-job suspend *Auftragsname*

mfwkadm pm-job resume *Auftragsname*

Überwachung des Betriebsstatus

mfwkadm opstat-job observable-classes

mfwkadm opstat-job observable-objects [class=*Objektklasse*] [domain= *Objektdomäne*]

mfwkadm opstat-job observable-attributes class= *Objektklasse*

mfwkadm opstat-job list

mfwkadm opstat-job info *Auftragsname*

mfwkadm opstat-job create *Auftragsname*
granularity= *Ganzzahliger_Wert* object=*Objektname* [object=*Objektname* ...]

mfwkadm opstat-job delete *Auftragsname*

mfwkadm opstat-job suspend *Auftragsname*

mfwkadm opstat-job resume *Auftragsname*

Schwellenwertüberwachung

mfwkadm thrsh-job observable-classes

mfwkadm thrsh-job observable-objects [class=*Objektklasse*] [domain= *Objektdomäne*]

mfwkadm thrsh-job observable-attributes class=*Objektklasse*

mfwkadm thrsh-job list

mfwkadm thrsh-job info *Auftragsname*

mfwkadm thrsh-job create *Auftragsname*
granularity=*Ganzzahliger_Wert* attributeName=*Attributname*
attributeType=*Attribut-Typ* thresholdValue=*Schwellenwert*
thresholdOffset=*Abweichung* thresholdDirection=[RISING | FALLING]
object=*Objektname*

mfwkadm thrsh-job delete *Auftragsname*

mfwkadm thrsh-job suspend *Auftragsname*

mfwkadm thrsh-job resume *Auftragsname*

Beschreibung

Das Dienstprogramm `mfwkadm` ist die Befehlszeilenschnittstelle zum Verwalten des Monitoring Framework-Aggenten (auch als Knotenagent bezeichnet). Der Knotenagent wird innerhalb des Common Agent Container ausgeführt. Mit dem Dienstprogramm `mfwkadm` können Sie

Knotenagenten starten und stoppen sowie die von einem Knotenagent ausgeführten Überwachungsaufträge verwalten. Der Befehl muss auf dem gleichen Host ausgeführt werden, auf dem der Knotenagent läuft. Die Reihenfolge der hier aufgeführten Befehlsargumente ist einzuhalten.

Zum Ändern der Sprache der ausgegebenen Meldungen müssen Sie die Umgebungsvariable `LC_MESSAGE` auf Ihre *Sprachumgebung* setzen. Der Befehl `mfwkadm` verwendet die in der Datei `JesmfMessages_` *Sprachumgebung*`.pm` enthaltenen Meldungen. Diese Datei befindet sich im Verzeichnis `lib/resources`. Falls für die gewünschte Sprachumgebung keine Meldungsdatei vorhanden ist oder keine Sprachumgebung angegeben wurde, verwendet der Befehl `mfwkadm` die Meldungen in der Standard-Meldungsdatei `JesmfMessages.pm`.

Das Dienstprogramm `mfwkadm` besitzt die folgenden Unterbefehle. Für Befehle, die mit einem Sternchen (*) gekennzeichnet sind, muss der Common Agent Container laufen und der Knotenagent muss geladen sein:

- `start`
- `stop`
- `restart`
- `list-params` (*)
- `list-modules` (*)
- `info` (*)
- `pm-job` (*)
- `opstat-job` (*)
- `thrsh-job` (*)

Je nach Anzahl der zu ladenden Common Agent Container-Module kann es sein, dass zwischen dem Starten des Knotenagenten und der Verfügbarkeit des Dienstprogramms `mfwkadm` eine Zeitverzögerung von einigen Sekunden bis hin zu einigen Minuten auftreten kann.. In diesem Zeitraum schlagen Befehle mit einer explizit ausgegebenen Meldung fehl.

Optionen

Die folgenden Optionen werden unterstützt.

`--help`	Anzeigen einer Syntaxübersicht.

Unterbefehle

`start`
: Starten des Monitoring Framework-Knotenagenten und der zugehörigen Produktkomponenten ohne Stoppen des Common Agent Container.

Diese Aktion stellt zuerst den Knotenagenten und dann die zugehörigen Produktkomponenten im Common Agent Container bereit. Bei dieser Funktion handelt es sich um eine Schicht über den Unterbefehlen `lock` und `undeploy` des Dienstprogramms `cacaoadm`.

Der Unterbefehl `start` startet nur den Knotenagenten und die zum Monitoring Framework gehörenden Java ES-Komponentenmodule. Komponentenmodule tragen das Präfix `com.sun.cmm`.

Sicherheit: Der Unterbefehl `start` kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat. Andernfalls wird eine Fehlermeldung angezeigt, die der folgenden Meldung ähnelt:

```
Error occured in mfwkadm
Problem running /usr/sbin/cacaoadm unlock com.sun.mfwk 2>&1.
Stdout/Stderr: This command must be run by user: [root].
```

`stop`

Stoppen des Monitoring Framework-Knotenagenten und seiner zugehörigen Java ES-Komponentenmodule im Common Agent Container.

Diese Aktion stoppt zunächst alle im Common Agent Container bereitgestellten Java ES-Komponent'enmodule und anschließend den Knotenagent. Bei dieser Funktion handelt es sich um eine Schicht über den Unterbefehlen `lock` und `undeploy` des Dienstprogramms `cacaoadm`.

Der Unterbefehl `stop` stoppt nur die Java ES-Komponentenmodule, die zum Monitoring Framework und dann dem Knotenagent selbst gehören. Komponentenmodule tragen das Präfix `com.sun.cmm`.

Sicherheit: Der Unterbefehl `stopt` kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat. Andernfalls wird eine Fehlermeldung angezeigt, die der folgenden Meldung ähnelt:

```
Error occured in mfwkadm
Problem running /usr/sbin/cacaoadm unlock com.sun.mfwk 2>&1.
Stdout/Stderr: This command must be run by user: [root].
```

`restart`

Neustart des Monitoring Framework-Knotenagenten und seiner zugehörigen Java ES-Komponentenmodule im Common Agent Container.

Diese Aktion versucht, den Knotenagenten und die zugehörigen Module im Common Agent Container zunächst zu stoppen und dann wieder neu zu starten. Dies geschieht in der gleichen Weise wie bei den Unterbefehlen `stop` und `start`.

Sicherheit: Der Unterbefehl `restart` kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat. Andernfalls wird eine Fehlermeldung angezeigt, die der folgenden Meldung ähnelt:

```
Error occured in mfwkadm
Problem running //usr/sbin/cacaoadm unlock com.sun.mfwk 2>&1.
Stdout/Stderr: This command must be run by user: [root].
```

`list-params`
Auflisten aller Konfigurationsparameter für einen Monitoring Framework-Knotenagenten.

Sicherheit: Für diesen Befehl existiert keine Benutzerbeschränkung.

`list-modules`
Anzeigen einer Liste der Produktkomponenten, die das Common Monitoring Model (CMM) implementieren und im Common Agent Container geladen sind. Dieser Unterbefehl listet auch alle laufenden Instanzen jeder installierten Java ES-Komponente auf. Jede Komponente kann null, eine oder mehrere ausgeführte Instanzen besitzen.

Sicherheit: Wenn Benutzer, die den Common Agent Container nicht gestartet haben, diesen Befehl ausführen, enthält die Liste installierter Java ES-Komponenten keine Komponenteninstanzen.

`info` *Ausgeführte_Instanz*
Anzeigen von Informationen für die im Parameter *Ausgeführte_Instanz* bezeichnete Instanz. Der Wert in *Ausgeführte_Instanz* muss einer in der Ausgabe des Unterbefehls `list-modules` aufgeführten Instanz entsprechen. Zu den angezeigten Informationen gehören:

- Für jeden Überwachungsauftragstyp alle zur laufenden Instanz gehörenden überwachbaren Objekte (sortiert nach Klassennamen). Zu den überwachbaren Objekten zählen all jene, für die Sie mithilfe der Unterbefehle `pm-job`, `opstat-job` oder `thrsh-job` Aufträge zur Überwachung der Leistung, des Betriebsstatus oder eines Schwellenwerts erstellen können.
- Für jede Klasse überwachbarer Objekte alle ihre überwachbaren Attribute einschließlich des Namen und Typs jedes Attributs.

Sicherheit: Wenn Benutzer, die den Common Agent Container nicht gestartet haben, diesen Befehl ausführen, werden keine Informationen angezeigt.

Leistungsüberwachung

`pm-job observable-classes`
Anzeigen einer Liste aller gegenwärtig überwachbaren Objektklassen, für die Aufträge zur Leistungsüberwachung erstellt werden können.

`pm-job observable-objects` [class= *Objektklasse*] [domain=*Objektdomäne*]
Anzeigen einer Liste aller gegenwärtig überwachbaren Objekte, für die Aufträge zur Leistungsüberwachung erstellt werden können. Standardmäßig werden alle Objekte überwachbarer Klassen in allen Domänen aufgeführt. Die Objektliste ist nach Klassennamen sortiert.

class=*Objektklasse*
Die Angabe des optionalen Parameters *Objektklasse* beschränkt die Ausgabe auf überwachbare Objekte der angegebenen Klasse. Der Wert des Parameters *Objektklasse* muss eine der vom Unterbefehl `pm-job observable-classes` aufgelisteten Klassen sein.

domain=*Objektdomäne*
Die Angabe des optionalen Parameters *Objektdomäne* beschränkt die Ausgabe auf überwachbare Objekte der angegebenen Domäne. Unter der Domäne eines Objekts versteht man die Zeichenkette vor dem Doppelpunkt (:) in einem Objektnamen.

`pm-job observable-attributes class=`*Objektklasse*
Anzeigen einer Liste aller überwachbaren Attribute der angegebenen *Objektklasse*. Attribute werden mit Namen und Typ angezeigt. Der Wert des Parameters *Objektklasse* muss eine Klasse sein, die Leistungsüberwachung unterstützt und somit vom Unterbefehl `pm-job observable-classes` ausgegeben wird.

`pm-job list`
Anzeigen einer Liste aller gegenwärtig definierten Aufträge zur Leistungsüberwachung. Aufträge werden für das jeweilige Objekt ausgegeben, für das ein Leistungsüberwachungsauftrag definiert ist, und die entsprechenden Objekte sind nach Klassennamen sortiert. Die für jeden Auftrag angezeigten Informationen sind die gleichen, die auch vom Unterbefehl `pm-job info` ausgegeben werden.

Sicherheit: Wenn Benutzer, die den Common Agent Container nicht gestartet haben, diesen Befehl ausführen, werden keine Aufträge angezeigt.

`pm-job info` *Auftragsname*
Anzeigen ausführlicher Informationen zu dem von *Auftragsname* bezeichneten Leistungsüberwachungsauftrag. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `pm-job list` ausgegebenen Liste entsprechen. Dieser Unterbefehl zeigt die folgenden Informationen an:

- Der Name des Auftrags zur Leistungsüberwachung.
- Der Typ des Auftrags zur Leistungsüberwachung: "by object" oder "by class.". Aufträge vom Typ "by object" überwachen bezeichnete Objektinstanzen, während Aufträge vom Typ "by class" jede Instanz einer Objektklasse überwachen. Bitte beachten Sie, dass Aufträge vom Typ "by class" mit dem Dienstprogramm `mfwkadm` nicht erstellt werden können.
- Der Status des Auftrags zur Leistungsüberwachung. "active on-duty", "active off-duty" oder "suspended". Ein Auftrag mit dem Status "active on-duty" ist gegenwärtig planmäßig aktiv und erfasst Daten. Ein Auftrag mit dem Status "active off-duty" läuft zwar, erfasst jedoch keine Daten, da er gegenwärtig gemäß dem Zeitplan nicht aktiv ist. Ein Auftrag mit dem Status "suspended" läuft nicht und erfasst keine Daten. Mit den Unterbefehlen `pm-job suspend` und `pm-job resume` können Sie den Laufzeitstatus eines Auftrags zur Leistungsüberwachung ändern.

- Die Granularität (Erfassungshäufigkeit) des betreffenden Auftrags zur Leistungsüberwachung (in Sekunden). Dieser Parameter gibt an, wie häufig der Auftrag Daten erfasst.
- Der Protokollierungszeitraum des Überwachungsauftrags. Der Protokollierungszeitraum multipliziert mit der Granularität ergibt die Benachrichtigungshäufigkeit. Wenn die Granularität beispielsweise 10 Sekunden und der Protokollierungszeitraum 6 Sekunden beträgt, erfasst dieser Auftrag alle 10 Sekunden Daten und sendet alle 60 Sekunden eine Benachrichtigung, die sechs Protokolle enthält (10*6). Wenn der Auftrag Protokollierungsdaten auch in eine Datei schreibt, sendet er alle 60 Sekunden ein Ereignis, das die Namen und Speicherorte der sechs erstellten Dateien enthält.
- Protokolliert der Leistungsüberwachungsauftrag nach Ereignissen? Das bedeutet, das die Ergebnisse des Leistungsüberwachungsauftrags in diesem Fall als Benachrichtigungen an einen registrierten Client gesendet werden.
- Protokolliert der Leistungsüberwachungsauftrag in eine Datei? Das bedeutet, das die Ergebnisse des Leistungsüberwachungsauftrags in diesem Fall in eine Datei geschrieben werden und nur die Dateinamen an einen registrierten Client gesendet werden.
- Das Protokollformat des Leistungsüberwachungsauftrags (immer `XML`).
- Der Zeitplan für den Leistungsüberwachungsauftrag. Der Zeitplan gibt an, an welchen Tagen und zu welcher Uhrzeit ein Auftrag aktiv sein und Daten erfassen soll. Für

"by-object"-Aufträge:

- Die Liste überwachter Objekte (nach Namen geordnet).
- Wenn nur eine Untermenge überwachbarer Attribute angegeben ist, werden die überwachten Attribute der überwachten Objekte nach Namen und Typen aufgeführt.

Für "by-class"-Aufträge:

- Die Liste überwachter Klassen (nach Namen geordnet).
- Wenn nur eine Untermenge überwachbarer Attribute angegeben ist, werden die überwachten Attribute der überwachten Klassen nach Namen und Typen aufgeführt. Diese Attribute sind für alle Klassen gleich.

Sicherheit: Wenn Benutzer, die den Common Agent Container nicht gestartet haben, diesen Befehl ausführen, werden keine Informationen angezeigt.

`pm-job create` *Auftragsname* granularity= *Ganzzahliger_Wert* object=*Objektname* [object= *Objektname* ...]

Erstellt für ein Objekt bzw. mehrere Objekte einen neuen Leistungsüberwachungsauftrag. Der Befehl `mfwkadm` kann keine Aufträge vom Typ "by class" erstellen. Beim Erstellen von Leistungsüberwachungsaufträgen können die folgenden Parameter eingestellt werden:

Auftragsname

Eine Zeichenkette, die den Leistungsüberwachungsauftrag eindeutig spezifiziert. Der Wert des Parameters *Auftragsname* darf nicht bereits von einem anderen Leistungsüberwachungsauftrag verwendet werden.

granularity=*Ganzzahliger_Wert*
Die Zeit (in Sekunden), die zwischen zwei Datenerfassungszeitpunkten vergeht , wenn der Auftrag den Status “active on-duty” besitzt. Die Granularität kann beispielsweise 300 Sekunden (5 Minuten), 900 Sekunden (15 Minuten), 1800 Sekunden (halbstündlich) oder 3600 Sekunden (stündlich) betragen. In den meisten Fällen reicht eine Granularität von 300 Sekunden aus. Manchmal ist es jedoch sinnvoller, Daten nicht so oft zu erfassen

object=*Objektname* [object= *Objektname* ...]
Ein bzw. mehrere überwachbare Objekte, für die der betreffende Leistungsüberwachungsauftrag Daten erfasst und protokolliert. Der Wert des Parameters *Objektname* muss in der von den Unterbefehlen `pm-job list` bzw. `pm-job observable-objects` ausgegebenen Objektliste enthalten sein. Durch Angabe mehrerer Werte für “object= *Objektname*” wird ein einziger Leistungsüberwachungsauftrag erstellt, der mehrere Objekte überwacht.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

`pm-job delete` *Auftragsname*
Löschen des Leistungsüberwachungsauftrags *Auftragsname*. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `pm-job list` ausgegebenen Liste entsprechen.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

`pm-job suspend` *Auftragsname*
Anhalten des Leistungsüberwachungsauftrags *Auftragsname*. Angehaltene Aufträge sind nicht aktiv und erfassen unabhängig von ihrem Zeitplan keine Daten. Der Auftrag bleibt jedoch erhalten und kann mithilfe des Unterbefehls `pm-job resume` wieder aktiviert werden. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `pm-job list` ausgegebenen Liste entsprechen.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

`pm-job resume` *Auftragsname*
Fortsetzen des Leistungsüberwachungsauftrags *Auftragsname*. Ein fortgesetzter Auftrag erfasst wieder Daten und protokolliert diese gemäß dem Zeitplan. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `pm-job list` ausgegebenen Liste entsprechen. Dies ist das Gegenstück zum Unterbefehl `pm-job suspend`.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

Überwachung des Betriebsstatus

`opstat-job observable-classes`

Anzeigen einer Liste aller gegenwärtig überwachbaren Objektklassen, für die Aufträge zur Betriebsstatusüberwachung erstellt werden können.

`opstat-job observable-objects` [class= *Objektklasse*] [domain=*Objektdomäne*]

Anzeigen einer Liste aller gegenwärtig überwachbaren Objekte, für die Aufträge zur Betriebsstatusüberwachung erstellt werden können. Standardmäßig werden alle Objekte überwachbarer Klassen in allen Domänen aufgeführt. Die Objektliste ist nach Klassennamen sortiert.

class=*Objektklasse*

Die Angabe des optionalen Parameters *Objektklasse* beschränkt die Ausgabe auf überwachbare Objekte der angegebenen Klasse. Der Wert des Parameters *Objektklasse* muss eine der vom Unterbefehl `opstat-job observable-classes` aufgelisteten Klassen sein.

domain=*Objektdomäne*

Die Angabe des optionalen Parameters *Objektdomäne* beschränkt die Ausgabe auf überwachbare Objekte der angegebenen Domäne. Unter der Domäne eines Objekts versteht man die Zeichenkette vor dem Doppelpunkt (:) im Objektnamen.

`opstat-job observable-attributes class=`*Objektklasse*

Anzeigen einer Liste aller überwachbaren Attribute der angegebenen *Objektklasse*. Attribute werden mit Namen und Typ angezeigt. Der Wert des Parameters *Objektklasse* muss eine der vom Unterbefehl `opstat-job observable-classes` aufgelisteten Klassen sein.

`opstat-job list`

Anzeigen einer Liste aller gegenwärtig definierten Aufträge zur Betriebsstatusüberwachung. Aufträge werden für das jeweilige Objekt ausgegeben, für das ein Auftrag zur Betriebsstatusüberwachung definiert ist, und die entsprechenden Objekte sind nach Klassennamen sortiert. Die für jeden Auftrag angezeigten Informationen sind die gleichen, die auch vom Unterbefehl `opstat-job info` ausgegeben werden.

Sicherheit: Wenn Benutzer, die den Common Agent Container nicht gestartet haben, diesen Befehl ausführen, werden keine Aufträge angezeigt.

`opstat-job info` *Auftragsname*

Anzeigen ausführlicher Informationen zu dem von *Auftragsname* bezeichneten Auftrag zur Betriebsstatusüberwachung. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `opstat-job list` ausgegebenen Liste entsprechen. Dieser Unterbefehl zeigt die folgenden Informationen an:

- Der Name des Auftrags zur Betriebsstatusüberwachung.
- Der Typ des Auftrags zur Betriebsstatusüberwachung: “by object” oder “by class.”. Aufträge vom Typ “by object” überwachen bezeichnete Objektinstanzen, während Aufträge vom Typ “by class” jede Instanz einer Objektklasse überwachen. Bitte beachten Sie, dass Aufträge vom Typ “by class” mit dem Dienstprogramm `mfwkadm` nicht erstellt werden können.
- Der Status des Auftrags zur Betriebsstatusüberwachung: “active on-duty”, “active off-duty” oder “suspended”. Ein Auftrag mit dem Status “active on-duty” ist gegenwärtig planmäßig aktiv und erfasst Daten. Ein Auftrag mit dem Status “active off-duty” läuft zwar, erfasst jedoch keine Daten, da er gegenwärtig gemäß dem Zeitplan nicht aktiv ist. Ein Auftrag mit dem Status “suspended” läuft nicht und erfasst keine Daten. Mit den Unterbefehlen `opstat-job suspend` und `opstat-job resume` können Sie den Laufzeitstatus eines Auftrags zur Betriebsstatusüberwachung ändern.
- Die Granularität (Erfassungshäufigkeit) des betreffenden Auftrags zur Betriebsstatusüberwachung (in Sekunden). Dieser Parameter gibt an, wie häufig der Auftrag Daten erfasst.
- Protokolliert der Auftrag zur Betriebsstatusüberwachung nach Ereignissen? Das bedeutet, das die Ergebnisse des Auftrags zur Betriebsstatusüberwachung in diesem Fall als Benachrichtigungen an einen registrierten Client gesendet werden.
- Protokolliert der Auftrag zur Betriebsstatusüberwachung in eine Datei? Das bedeutet, das die Ergebnisse des Auftrags zur Betriebsstatusüberwachung in diesem Fall in eine Datei geschrieben werden und nur die Dateinamen an einen registrierten Client gesendet werden.
- Das Protokollformat des Auftrags zur Betriebsstatusüberwachung (immer `XML`).
- Der Zeitplan des Auftrags zur Betriebsstatusüberwachung: Der Zeitplan gibt an, an welchen Tagen und zu welcher Uhrzeit ein Auftrag aktiv sein und Daten erfassen soll.
- Aufträge vom Typ “by-object”: Die Liste aller überwachten Objekte (nach Namen geordnet).
- Aufträge vom Typ “by-class”: Die Liste aller überwachten Klassen (nach Namen geordnet).

Sicherheit: Wenn Benutzer, die den Common Agent Container nicht gestartet haben, diesen Befehl ausführen, werden keine Informationen angezeigt.

`opstat-job` `create` *Auftragsname* granularity= *Ganzzahliger_Wert* object=*Objektname* [object= *Objektname* ...]

Erstellt für ein Objekt bzw. mehrere Objekte einen neuen Auftrag zur Betriebsstatusüberwachung. Der Befehl `mfwkadm` kann keine Aufträge vom Typ "by class" erstellen. Beim Erstellen von Leistungsüberwachungsaufträgen können die folgenden Parameter eingestellt werden:

Auftragsname
Eine Zeichenkette, die den Auftrag zur Betriebsstatusüberwachung eindeutig spezifiziert. Der Wert des Parameters *Auftragsname* darf nicht bereits von einem anderen Auftrag zur Betriebsstatusüberwachung verwendet werden.

granularity=*Ganzzahliger_Wert*
Die Zeit (in Sekunden), die zwischen zwei Datenerfassungszeitpunkten vergeht , wenn der Auftrag den Status "active on-duty" besitzt.

object=*Objektname* [object= *Objektname* ...]
Ein bzw. mehrere überwachbare Objekte, für die der betreffende Auftrag zur Betriebsstatusüberwachung Daten erfasst und protokolliert. Der Wert des Parameters *Objektname* muss in der von den Unterbefehlen `opstat-job list` bzw. `opstat-job observable-objects` ausgegebenen Objektliste enthalten sein. Durch Angabe mehrerer Werte für "object= *Objektname*" wird ein einziger Auftrag zur Betriebsstatusüberwachung erstellt, der mehrere Objekte überwacht.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

`opstat-job` `delete` *Auftragsname*

Löschen des Auftrags zur Betriebsstatusüberwachung *Auftragsname*. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `opstat-job list` ausgegebenen Liste entsprechen.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

`opstat-job` `suspend` *Auftragsname*

Anhalten des Auftrags zur Betriebsstatusüberwachung *Auftragsname*. Angehaltene Aufträge sind nicht aktiv und erfassen unabhängig von ihrem Zeitplan keine Daten. Der Auftrag bleibt jedoch erhalten und kann mithilfe des Unterbefehls `opstat-job resume` wieder aktiviert werden. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `opstat-job list` ausgegebenen Liste entsprechen.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

`opstat-job` `resume` *Auftragsname*

Fortsetzen des Auftrags zur Betriebsstatusüberwachung *Auftragsname*. Ein fortgesetzter Auftrag erfasst wieder Daten und protokolliert diese gemäß dem Zeitplan. Der Wert des

Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `opstat-job list` ausgegebenen Liste entsprechen. Dies ist das Gegenstück zum Unterbefehl `opstat-job suspend`.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

Schwellenwertüberwachung

`thrsh-job observable-classes`
Anzeigen einer Liste aller gegenwärtig überwachbaren Objektklassen, für die Schwellenwertüberwachungsaufträge erstellt werden können.

`thrsh-job observable-objects` [class= *Objektklasse*] [domain=*Objektdomäne*]
Anzeigen einer Liste aller gegenwärtig überwachbaren Objekte, für die Schwellenwertüberwachungsaufträge erstellt werden können. Standardmäßig werden alle Objekte überwachbarer Klassen in allen Domänen aufgeführt. Die Objektliste ist nach Klassennamen sortiert.

class=*Objektklasse*
Die Angabe des optionalen Parameters *Objektklasse* beschränkt die Ausgabe auf überwachbare Objekte der angegebenen Klasse. Der Wert des Parameters *Objektklasse* muss eine der vom Unterbefehl `thrsh-job observable-classes` aufgelisteten Klassen sein.

domain=*Objektdomäne*
Die Angabe des optionalen Parameters *Objektdomäne* beschränkt die Ausgabe auf überwachbare Objekte der angegebenen Domäne. Unter der Domäne eines Objekts versteht man die Zeichenkette vor dem Doppelpunkt (:) im Objektnamen.

`thrsh-job observable-attributes class=`*Objektklasse*
Anzeigen einer Liste aller überwachbaren Attribute der angegebenen *Objektklasse*. Attribute werden mit Namen und Typ angezeigt. Der Wert des Parameters *Objektklasse* muss eine der vom Unterbefehl `thrsh-job observable-classes` aufgelisteten Klassen sein.

`thrsh-job list`
Anzeigen einer Liste aller gegenwärtig definierten Schwellenwertüberwachungsaufträge. Aufträge werden für das jeweilige Objekt ausgegeben, für das ein Schwellenwertüberwachungsauftrag definiert ist, und die entsprechenden Objekte sind nach Klassennamen sortiert. Die für jeden Auftrag angezeigten Informationen sind die gleichen, die auch vom Unterbefehl `thrsh-job info` ausgegeben werden.

Sicherheit: Wenn Benutzer, die den Common Agent Container nicht gestartet haben, diesen Befehl ausführen, werden keine Aufträge angezeigt.

`thrsh-job info` *Auftragsname*

Anzeigen ausführlicher Informationen zu dem von *Auftragsname* bezeichneten Schwellenwertüberwachungsauftrag. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `thrsh-job list` ausgegebenen Liste entsprechen. Dieser Unterbefehl zeigt die folgenden Informationen an:

- Der Name des Auftrags zur Schwellenwertüberwachung.
- Die Anzahl der ausgeführten Schwellenwertüberwachungsaufträge. In dieser Version sind lediglich einfache Schwellenwertüberwachungsaufträge möglich, die ein Attribut bzw. ein Objekt überwachen.
- Der Status des Auftrags zur Schwellenwertüberwachung. “active on-duty”, “active off-duty” oder “suspended”. Ein Auftrag mit dem Status “active on-duty” ist gegenwärtig planmäßig aktiv und erfasst Daten. Ein Auftrag mit dem Status “active off-duty” läuft zwar, erfasst jedoch keine Daten, da er gegenwärtig gemäß dem Zeitplan nicht aktiv ist. Ein Auftrag mit dem Status “suspended” läuft nicht und erfasst keine Daten. Mit den Unterbefehlen `thrsh-job suspend` und `thrsh-job resume` können Sie den Laufzeitstatus eines Schwellenwertüberwachungsauftrags ändern.
- Die Granularität (Erfassungshäufigkeit) des betreffenden Schwellenwertüberwachungsauftrags (in Sekunden). Dieser Parameter gibt an, wie häufig der Auftrag Daten erfasst.
- Der Zeitplan des Auftrags zur Schwellenwertüberwachung. Der Zeitplan gibt an, an welchen Tagen und zu welcher Uhrzeit ein Auftrag aktiv sein und Daten erfassen soll.
- Die Alarmkonfiguration des Schwellenwertüberwachungsauftrags. Das ist der Alarm, der ausgelöst wird, wenn der erfasste Wert des überwachten Objekts den festgelegten Schwellenwert über- bzw. unterschreitet. Es werden der Typ und der Schweregrad des Alarms angezeigt.
- Das überwachte Objekt des Schwellenwertüberwachungsauftrags.
- Der Name des Attributs, das durch den Schwellenwertüberwachungsauftrag überwacht wird.
- Der Wert des Schwellenwerts, der einen Alarm auslöst.
- Die Richtung des Werteverlaufs, die einen Schwellenwertalarm auslösen soll (`RISING` oder `FALLING`, steigend bzw. fallend).
- Die Abweichungstoleranz des Schwellenwerts. Bei der Richtung `RISING` wird erst wieder ein Alarm ausgelöst, wenn der überwachte Attributwert kleiner als der *Schwellenwert-Abweichung* ist. Bei der Richtung `FALLING` wird erst wieder ein Alarm ausgelöst, wenn der überwachte Attributwert größer als der *Schwellenwert+Abweichung* ist. Dieses Verhalten gilt auch dann, wenn die Abweichung null ist.

Sicherheit: Wenn Benutzer, die den Common Agent Container nicht gestartet haben, diesen Befehl ausführen, werden keine Informationen angezeigt.

`thrsh-job create` *Auftragsname* object= *Objektname* granularity=*Ganzzahliger_Wert* attributeName= *Attributname* attributeType=*Attribut-Typ* thresholdValue= *Schwellenwert* thresholdOffset=*Abweichung* thresholdDirection= `[RISING|FALLING]`

Erstellt einen neuen Schwellenwertüberwachungsauftrag, der ein Attribut eines einzelnen Objekts überwacht. Beim Erstellen von Schwellenwertüberwachungsaufträgen können die folgenden Parameter eingestellt werden:

Auftragsname
: Eine Zeichenkette, die den Schwellenwertüberwachungsauftrag eindeutig spezifiziert. Der Wert des Parameters *Auftragsname* darf nicht bereits von einem anderen Schwellenwertüberwachungsauftrag verwendet werden.

object=*Objektname*
: Das überwachbare Objekt, für das der Schwellenwertüberwachungsauftrag die Attributwerte zum Vergleich mit dem Schwellenwert erfasst. Der Wert des Parameters *Objektname* muss in der von den Unterbefehlen `thrsh-job list` bzw. `thrsh-job observable-objects` ausgegebenen Objektliste enthalten sein.

granularity=*Ganzzahliger_Wert*
: Die Zeit (in Sekunden), die zwischen zwei Erfassungszeitpunkten eines Attributwertes vergeht , wenn der Auftrag den Status “active on-duty” besitzt.

attributeName=*Attributname*
: Der Name des Attributs, dessen Werte der Schwellenwertüberwachungsauftrag erfasst und die mit dem Schwellenwert verglichen werden. Der Wert des Parameters *Attributname* muss in der von den Unterbefehlen `thrsh-job info` bzw. `thrsh-job observable-attributes` ausgegebenen Objektliste enthalten sein.

attributeType=*Attribute-Typ*
: Der Typ des zu überwachenden Attributs. Der Wert des Parameters *Attribut-Typ* muss in der von den Unterbefehlen `thrsh-job info` bzw. `thrsh-job observable-attributes` ausgegebenen Objektliste enthalten sein.

thresholdValue=*Schwellenwert*
: Der Wert des überwachten Attributs, bei dem der Schwellenwertüberwachungsauftrag einen Alarm auslösen soll, wenn er in der im Parameter `thresholdDirection` angegebenen Richtung über- bzw. unterschritten wird.

thresholdOffset=*Abweichung*
: Die *Abweichung* legt für den Schwellenwertüberwachungsauftrag die Abweichungstoleranz für das Auslösen aufeinander folgender Alarme fest. Die *Abweichung* muss null oder positiv sein. Nach dem Auslösen eines Alarms wird erst wieder ein neuer Alarm ausgelöst, wenn der Wert des überwachten Attributs den im Parameter *Abweichung* festgelegten Wert über- bzw. unterschreitet (in Abhängigkeit vom Wert des Parameters `thresholdDirection`).

thresholdDirection=`[RISING|FALLING]`
: Bei der Richtung `RISING` wird erst wieder ein Alarm ausgelöst, wenn der überwachte Attributwert kleiner als der *Schwellenwert-Abweichung* ist. Bei der Richtung `FALLING`

wird erst wieder ein Alarm ausgelöst, wenn der überwachte Attributwert größer als der *Schwellenwert+Abweichung* ist. Dieses Verhalten gilt auch dann, wenn die *Abweichung* null ist.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

`thrsh-job delete` *Auftragsname*
Löschen des Schwellenwertüberwachungsauftrags *Auftragsname*. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `thrsh-job list` ausgegebenen Liste entsprechen.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

`thrsh-job suspend` *Auftragsname*
Anhalten des Schwellenwertüberwachungsauftrags *Auftragsname*. Angehaltene Aufträge sind nicht aktiv und erfassen unabhängig von ihrem Zeitplan keine Daten. Der Auftrag bleibt jedoch erhalten und kann mithilfe des Unterbefehls `thrsh-job resume` wieder aktiviert werden. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `thrsh-job list` ausgegebenen Liste entsprechen.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

`thrsh-job resume` *Auftragsname*
Fortsetzen des Schwellenwertüberwachungsauftrags *Auftragsname*. Ein fortgesetzter Auftrag erfasst wieder Daten und protokolliert diese gemäß dem Zeitplan. Der Wert des Parameters *Auftragsname* muss einem Auftrag aus der vom Unterbefehl `thrsh-job list` ausgegebenen Liste entsprechen. Dies ist das Gegenstück zum Unterbefehl `thrsh-job suspend`.

Sicherheit: Dieser Unterbefehl kann nur von dem Benutzer ausgeführt werden, der den Common Agent Container gestartet hat.

Beispiele

Im folgenden hypothetischen Beispiel wird die Verwendung des Dienstprogramms `mfwkadm` mit seinen Optionen und Unterbefehlen demonstriert.

Der Unterbefehl `list-modules` zeigt die Java ES-Komponenteninstanzen auf dem aktuellen Host und die entsprechenden Module im Common Agent Container an. Im folgenden Beispiel sind zwei installierte Komponenten (Directory Server mit zwei ausgeführten Instanzen sowie Web Server mit einer ausgeführten Instanz) aufgeführt.

```
$ mfwkadm list-modules

Installed products and their running instances:
==============================================

Instances for installed product: com.sun.cmm.ds:collectionID=/opt/SUNWdsee/ds6,
name=Sun Java(TM) System Directory Server,type=CMM_InstalledProduct
-------------------------------

No instance.

Instances for installed product: com.sun.cmm.ws:collectionID=/var/opt/SUNWwbsvr7,
name=WebServer,type=CMM_InstalledProduct
-------------------------------

/wsPrefix/com.sun.cmm.ws:name=https-hostname.example.com,type=CMM_ApplicationSystem
```

Der folgende Unterbefehl `info` zeigt überwachbare Objekte in der Web Server-Instanz mit den Klassen und Attribute für jeden Objekttyp an.

```
$ mfwkadm info /wsPrefix/com.sun.cmm.ws:name=https-hostname.example.com,\\
type=CMM_ApplicationSystem

Information about running instance: /wsPrefix/com.sun.cmm.ws:
name=https-hostname.example.com,type=CMM_ApplicationSystem
==================================

Observable objects for performance jobs:
---------------------------------------

+ Objects of class: com.sun.cmm.settings.CMM_ApplicationSystemSetting

        /wsPrefix/com.sun.cmm.ws:name=https-hostname.example.com-setting,
type=CMM_ApplicationSystemSetting

        Observable attributes:

        Caption [STRING]
        ConfigurationDirectory [STRING]
        CreationClassName [STRING]
        Description [STRING]
        DirectoryName [STRING]
        ElementName [STRING]
        InstanceID [STRING]
        Name [STRING]
        URL [STRING]

+ Objects of class: com.sun.cmm.settings.CMM_KeepAliveSetting
```

```
        /wsPrefix/com.sun.cmm.ws:name=process-1-keepalive-setting,
type=CMM_KeepAliveSetting

        Observable attributes:

        AllocationUnit [STRING]
        Caption [STRING]
        ConnectionsUpperBound [LONG]
        CreationClassName [STRING]
        Description [STRING]
        ElementName [STRING]
        InputUnit [STRING]
        InstanceID [STRING]
        LowerAllocationLimit [LONG]
        LowerInputLimit [LONG]
        LowerOutputLimit [LONG]
        Name [STRING]
        OtherAllocationUnit [STRING]
        OtherInputUnit [STRING]
        OtherLowerAllocationLimit [LONG]
        OtherLowerInputLimit [LONG]
        OtherLowerOutputLimit [LONG]
        OtherOutputUnit [STRING]
        OtherUpperAllocationLimit [LONG]
        OtherUpperInputLimit [LONG]
        OtherUpperOutputLimit [LONG]
        OutputUnit [STRING]
        QueuedUpperBound [LONG]
        SecondsTimeout [LONG]
        TimeoutUpperBound [LONG]
        UpperAllocationLimit [LONG]
        UpperInputLimit [LONG]
        UpperOutputLimit [LONG]
        ...
```

Der folgende Befehl gibt die Liste definierter Leistungsüberwachungsaufträge aus. In diesem Beispiel ist ein Leistungsüberwachungsauftrag namens `MeinLeistgUebAuftrag` definiert, der ein Objekt überwacht:

```
$ mfwkadm pm-job list

BY_OBJECTS performance jobs:
===========================

Performance job information for: MeinLeistgUebAuftrag
-------------------------------
```

```
Type:                 BY_OBJECTS
State:                ACTIVE_ON_DUTY
Granularity period:   30
Reporting period:     1
By event:             EVENT_SINGLE
By file:              EVENT_SINGLE
Report format:        XML
Schedule:
        Global start time: Immediately
        Global stop time: Forever
        Weekly schedule: Everyday
        Daily schedule: All day
Observed objects:
                /wsPrefix/com.sun.cmm.ws:name=virtualServer-hostname.example.com-
webApp-/-stats,type=CMM_VirtualServerWebModuleStats
Observed attributes:
                All available

BY_CLASSES performance jobs:
===========================

No jobs found.
```

Der folgende Befehl erstellt einen Auftrag zur Betriebsstatusüberwachung für zwei überwachbare Objekte aus den Unterbefehlen `opstat-job info` bzw. `opstat-job observable-objects`:

```
$ mfwkadm opstat-job create myOpStatJob granularity=60 \\
object=/wsPrefix/com.sun.cmm.ws:name=process-1,type=CMM_UnixProcess \\
object=/wsPrefix/com.sun.cmm.ws:name=process-1-DNSCache1,type=CMM_DnsCache
```

Der folgende Befehl hält den oben erstellten Auftrag an:

```
$ mfwkadm opstat-job suspend myOpStatJob
```

Der folgende Befehl zeigt die überwachbaren Klassen für potenzielle Schwellenwertüberwachungsaufträge an:

```
$ mfwkadm thrsh-job observable-classes

Threshold jobs observable classes:
=================================

com.sun.cmm.cim.CIM_ScopedSettingData
com.sun.cmm.cim.CIM_SettingData
com.sun.cmm.cim.CIM_StatisticalData
com.sun.cmm.cim.statistics.CIM_EthernetPortStatistics
com.sun.cmm.cim.statistics.CIM_NetworkPortStatistics
```

```
com.sun.cmm.cim.statistics.j2ee.CIM_J2eeJVMStats
com.sun.cmm.cim.statistics.j2ee.CIM_J2eeStatistic
com.sun.cmm.settings.CMM_ApplicationSystemSetting
com.sun.cmm.settings.CMM_KeepAliveSetting
com.sun.cmm.settings.CMM_QueueTimeoutSetting
com.sun.cmm.settings.CMM_RFC2788ApplicationSystemSetting
com.sun.cmm.settings.CMM_ScopedSettingData
com.sun.cmm.settings.CMM_SoftwareResourceSetting
com.sun.cmm.settings.CMM_SWRBufferSetting
com.sun.cmm.settings.CMM_SWRLimitSetting
com.sun.cmm.settings.CMM_SWRQueueSetting
com.sun.cmm.settings.CMM_VirtualServerSetting
com.sun.cmm.statistics.CMM_ApplicationSystemStats
com.sun.cmm.statistics.CMM_ApplicationSystemWatchdogStats
com.sun.cmm.statistics.CMM_ConnectionQueueStats
com.sun.cmm.statistics.CMM_DnsCacheStats
com.sun.cmm.statistics.CMM_EthernetPortStats
com.sun.cmm.statistics.CMM_FileCacheStats
com.sun.cmm.statistics.CMM_HTTPResponsesStats
com.sun.cmm.statistics.CMM_JVMJSR174ExtStats
com.sun.cmm.statistics.CMM_JVMJSR174Stats
com.sun.cmm.statistics.CMM_JVMStats
com.sun.cmm.statistics.CMM_NetworkPortStats
com.sun.cmm.statistics.CMM_OperatingSystemStats
com.sun.cmm.statistics.CMM_ProcessorStats
com.sun.cmm.statistics.CMM_ProcessStats
com.sun.cmm.statistics.CMM_QueueTimeoutStats
com.sun.cmm.statistics.CMM_RFC2788ApplicationTableStats
com.sun.cmm.statistics.CMM_ServiceStats
com.sun.cmm.statistics.CMM_SoftwareResourceStats
com.sun.cmm.statistics.CMM_SolarisEthernetPortStats
com.sun.cmm.statistics.CMM_SolarisNetworkPortStats
com.sun.cmm.statistics.CMM_SolarisOperatingSystemStats
com.sun.cmm.statistics.CMM_SolarisProcessorStats
com.sun.cmm.statistics.CMM_SolarisProcessorSysinfoStats
com.sun.cmm.statistics.CMM_SolarisProcessorVmStats
com.sun.cmm.statistics.CMM_Statistic
com.sun.cmm.statistics.CMM_SWRBufferStats
com.sun.cmm.statistics.CMM_SWRCacheStats
com.sun.cmm.statistics.CMM_SWRLimitStats
com.sun.cmm.statistics.CMM_SWRQueueStats
com.sun.cmm.statistics.CMM_UnixOperatingSystemStats
com.sun.cmm.statistics.CMM_UnixProcessStats
com.sun.cmm.statistics.CMM_VirtualServerWebModuleStats
com.sun.cmm.statistics.CMM_WebModuleStats
```

Der folgende Befehl gibt die überwachbaren Attribute für Schwellenwertüberwachungsaufträge aus, die die Objekte der Klasse `com.sun.cmm.statistics.CMM_SWRQueueStats` aus dem vorherigen Beispiel überwachen:

```
$ mfwkadm thrsh-job observable-attributes \\
class=com.sun.cmm.statistics.CMM_SWRQueueStats

Threshold jobs observable attributes:
====================================

Class: com.sun.cmm.statistics.CMM_SWRQueueStats

Attributes:

BufferSize [LONG]
EntriesCount [LONG]
EntriesHighWaterMark [LONG]
EntriesLowWaterMark [LONG]
EntriesTotal [LONG]
ErrorCount [INTEGER]
FailedOperations [LONG]
LowerLimit [LONG]
OperationsCount [LONG]
OtherLowerLimit [LONG]
OtherUpperLimit [LONG]
OverflowsCount [LONG]
QueuedCount [LONG]
QueuedHighWater [LONG]
SampleInterval [LONG]
TotalQueuedCount [LONG]
UpperLimit [LONG]
```

Der folgende Befehl ist ein weiteres Beispiel für die Auftragserstellung, diesmal für einen Schwellenwertüberwachungsauftrag:

```
$ mfwkadm thrsh-job create myThreshJob granularity=30 \\
object=/wsPrefix/com.sun.cmm.ws:name=process-1-threadPool-NativePool-stats,\\
type=CMM_SWRQueueStats attributeName=EntriesCount attributeType=LONG \\
thresholdValue=1000 thresholdOffset=10 thresholdDirection=RISING
```

Das folgende Beispiel zeigt die Ausgabe des Unterbefehls `thrsh-job info` den im vorigen Beispiel erstellten Schwellenwertüberwachungsauftrag:

```
$ mfwkadm thrsh-job info myThreshJob

Threshold job information for: myThreshJob
-----------------------------

Type:                SIMPLE
State:               ACTIVE_ON_DUTY
Granularity period:  30
Schedule:
```

```
        Global start time: Immediately
        Global stop time: Forever
        Weekly schedule: Everyday
        Daily schedule: All day
Alarm configuration:
                Type: QualityOfServiceAlarm
                Severity: INDETERMINATE
Threshold definition(s):
                Object: /wsPrefix/com.sun.cmm.ws:name=process-1-threadPool-
NativePool-stats,type=CMM_SWRQueueStats
                        Attribute: EntriesCount [LONG]
                        Value: 1000
                        Direction: RISING
                        Offset: 10
```

Beendigungsstatus

Bei Programmende werden die folgenden Werte zurückgegeben:

0 Erfolgreiche Kompilierung.

1 Ein Fehler ist aufgetreten.

Attribute

Attribut-Typ	Attributwert
Availability	SUNWmfwk
Interface Stability	Contract Private

Siehe auch:

cacao.5, cacaoadm.1m



Installation und Verwendung der Monitoring Console

Die Monitoring Console ist die internetbasierte Anwendung, die alle von der Instrumentierung erfassten Daten anzeigt. Sie baut auf einem Master-Agenten auf, der alle Werte und Alarmbenachrichtigungen von den einzelnen Knotenagenten sammelt und zusammenfasst.

Nach der Installation der Monitoring Console kann sie sicher in einem Browserfenster auf jedem beliebigen Host-Rechner oder sogar über das Internet (falls Ihre Firewall-Konfiguration dies zulässt) aufgerufen werden. Auf der grafischen Benutzeroberfläche können Sie die überwachten Werte in Echtzeit verfolgen, Alarme anzeigen und quittieren und Regeln zum Auslösen benutzerdefinierter Alarme erstellen.

Hinweis – Vor dem Ausführen einer Installation bzw. Konfiguration sollten Sie zunächst die *Sun Java Enterprise System 5 Versionshinweise für UNIX* durcharbeiten.

Dieses Kapitel enthält die folgenden Abschnitte:

Installation der Monitoring Console

Wegen der für den Master-Agenten geltenden Einschränkungen in dieser Version darf der Master-Agent nicht auf dem gleichen Host-Rechner wie die Knotenagenten laufen. Deswegen kann die Monitoring Console nicht auf dem gleichen Host-Rechner wie dem der überwachten Komponenten von Java ES, sondern auf einem eigenen Host-Rechner installiert werden, es sei denn, Sie haben Solaris Zones installiert. Weitere Informationen dazu finden Sie im folgenden Abschnitt „Installation der Monitoring Console in einer Solaris Zone" auf Seite 55

Durch die Installation der Monitoring Console wird die Monitoring Framework auch als eine Abhängigkeit für gemeinsam genutzte Komponenten installiert. Die Konsole benötigt zum Laden des Master-Agenten das Framework und den Common Agent Container, im Gegensatz zum Knotenagenten ist der Master-Agent jedoch nicht benutzerspezifisch konfigurierbar. Insbesondere sollte der Befehl `mfwkadm` nicht auf dem Host-Rechner bzw. der Zone, wo die Monitoring Console installiert ist, verwendet werden.

▼ Installation der Monitoring Console mit dem Java ES-Installationsprogramm

Wegen der für diese Beta-Version geltenden Einschränkungen müssen Sie die Monitoring Console auf einem Host-Rechner bzw. einer Solaris Zone installieren, wo keine anderen Java ES-Komponenten installiert sind. Deswegen ist die Monitoring Console die einzige Komponente, die bei diesem Vorgang installiert wird.

Bei diesem Vorgang wird die grafische Benutzeroberfläche des Installationsprogramms verwendet. Informationen zur Ausführung des Installationsprogramms in anderen Modi finden Sie in Kapitel 4, „Installation über die textbasierte Oberfläche” in *Sun Java Enterprise System 5 Installationshandbuch für UNIX* und Kapitel 5, „Installieren im stillen Modus” in *Sun Java Enterprise System 5 Installationshandbuch für UNIX*.

1. **Rufen Sie das Anwendungsprogramm `installer` aus dem Ihrer Plattform entsprechenden Verzeichnis in Java ES auf. Ausführlichere Informationen dazu finden Sie im Abschnitt „So beginnen Sie die Installation” in** *Sun Java Enterprise System 5 Installationshandbuch für UNIX***.**

2. **Nachdem Sie den Begrüßungsbildschirm geschlossen und die Lizenzbedingungen akzeptiert haben, müssen Sie entscheiden, ob Sie ein Upgrade oder eine Installation durchführen wollen. Klicken Sie auf "Neue Software installieren", und dann auf "Weiter".**

3. **Im Bildschirm zur Komponentenauswahl sollten Sie zur Installation nur die Sun Java System Monitoring Console auswählen. Klicken Sie auf "Weiter".**

4. **Das Installationsprogramm prüft die gemeinsam genutzten Komponenten auf erforderliche Upgrades. Klicken Sie auf "Weiter", wenn dies abgeschlossen ist.**

5. **Das Installationsprogramm überprüft jetzt die Systemvoraussetzungen. Falls auf dem Betriebssystem Patches zu installieren sind, müssen Sie die Installation abbrechen, die erforderlichen Patches auf dem System installieren und dann das Installationsprogramm erneut starten. Klicken Sie ansonsten auf "Weiter".**

6. **Klicken Sie im Bildschirm zur Auswahl des Konfigurationstyps auf "Jetzt konfigurieren" und im nächsten Konfigurationsbildschirm auf "Weiter".**

7. **Das Installationsprogramm ist nun zur Installation der Monitoring Console bereit. Klicken Sie auf "Weiter", um mit der Installation zu beginnen. Während der Installation können sie das Fenster zur Produktregistrierung öffnen, falls Sie Ihre Bereitstellung von Java ES noch nicht registriert haben.**

8. **Nach Abschluss der Installation können Sie die Installationszusammenfassung und die Protokolle durchgehen. Klicken Sie dann auf "Installation abgeschlossen", um das Installationsprogramm zu beenden.**

Nächste Schritte Gehen Sie jetzt zum Abschnitt „Konfigurieren der Monitoring Console" auf Seite 56.

▼ Installation der Monitoring Console in einer Solaris Zone

Mithilfe von Solaris Zones können Sie die Monitoring Console auf dem gleichen Host-Rechner wie andere Komponenten von Java ES installieren. Diese Komponenten befinden sich in der globalen Zone, und Sie erstellen eine lokale Sparse-Root-Zone, die als logischer Host für die Monitoring Console dient. Gehen Sie wie folgt vor.

1. **Installieren und konfigurieren Sie alle Java ES-Komponenten außer der Monitoring Console in der globalen Zone. Nehmen Sie alle nach der Installation erforderlichen Konfigurationen der ausgewählten Komponenten in der globalen Zone vor, sodass alle Serverinstanzen laufen.**

2. **Als Teil der Installation in der globalen Zone wird die Monitoring Framework als gemeinsam genutzte Komponente in der globalen Zone installiert. Führen Sie alle in Kapitel 2 beschriebenen Vorgänge aus, die für die von Ihnen installierten Komponenten gelten.**

3. **Erstellen Sie auf dem gleichen Host-Rechner eine lokale Sparse-Root-Zone als logischen Host für die Monitoring Console. Da es sich um eine Sparse-Root-Zone handelt, sollte das in** *mfwk-Basisverzeichnis* **installierte Monitoring Framework sichtbar sein (siehe „Standardpfade und Dateinamen" auf Seite 9).**

4. **Installieren Sie die Monitoring Console gemäß des in Abschnitt „Installation der Monitoring Console mit dem Java ES-Installationsprogramm" auf Seite 54 beschriebenen Vorgangs in der lokalen Sparse-Root-Zone.**

5. **Konfigurieren Sie das Monitoring Framework in der Sparse-Root-Zone mit den folgenden Befehlen:**

   ```
   cd mfwk-Basisverzeichnis/bin
   ./mfwksetup -i
   ```

   Dieser Befehl erstellt mithilfe der Dateien aus der globalen Zone die erforderlichen Konfigurationsdateien für Monitoring Framework in der lokalen Zone.

Nächste Schritte Gehen Sie jetzt zum Abschnitt „Konfigurieren der Monitoring Console” auf Seite 56.

▼ Konfigurieren der Monitoring Console

In diesem Abschnitt wird die Konfiguration der Monitoring Console auf einem einzelnen Host-Rechner beschrieben. Wenn Sie die Monitoring Console auf einem von einer Solaris Zone erstellten logischen Host installiert haben, sind die Befehle die gleichen, müssen jedoch im Dateisystem der betreffenden Zone ausgeführt werden.

1. **Mit dem Monitoring Framework können Sie den Master-Agenten mithilfe der folgenden Befehle initialisieren:**

   ```
   cd mfwk-Basisverzeichnis/bin
   ./masetup -i
   ```

2. **Starten Sie den Common Agent Container (`cacao`) mit dem folgenden Befehl neu:**

   ```
   cacaoadm restart
   ```

▼ Dekonfigurieren der Monitoring Console

Wenn Sie die Monitoring Console auf einem Host-Rechner, auf dem noch andere Komponenten installiert werden sollen, installieren und konfigurieren, können Sie diese Komponenten aufgrund eines Konflikts mit dem Monitoring Framework nicht überwachen. Damit Sie die neuen Komponenten mit einem Knotenagenten überwachen können, müssen Sie den Master-Agenten der Monitoring Console dekonfigurieren.

- **Melden Sie sich als root-Benutzer an und führen Sie zur Dekonfiguration der Monitoring Console die folgenden Befehle aus:**

  ```
  cacaoadm stop
  cacaoadm unregister-module com.sun.mfwk.masteragent.xml
  cacaoadm register-module /etc/mfwk-Basisverzeichnis/xml/com.sun.mfwk.xml
  cacaoadm restart
  ```

Struktur der installierten Verzeichnisse

Die Bezeichnung des auf Ihrem Betriebssystem installierten Pakets entnehmen Sie bitte aus Kapitel 5, „Liste der installierbaren Pakete” in *Sun Java Enterprise System 5 Installationsanweisungen für UNIX*. In der folgenden Tabelle sind die im Monitoring Console-Paket enthaltenen Verzeichnisse beschrieben. Der Parameter für das Standard-Installationsverzeichnis *MConsole-Basisverzeichnis* steht für die folgenden Verzeichnisse (siehe „Standardpfade und Dateinamen” auf Seite 9):

- Solaris: `/opt/SUNWjesmc`

- Linux: `/opt/sun/jesmc`

TABELLE 3–1 Verzeichnisse und Dateien, die von der Monitoring Console verwendet werden

Pfad	Beschreibung
MConsole-Basisverzeichnis`/WEB-INF/classes`	Servlet-Klassen für Internet-basierte Anwendungen
MConsole-Basisverzeichnis`/WEB-INF/lib`	JAR-Abhängigkeiten für Internet-basierte Anwendungen
MConsole-Basisverzeichnis`/WEB-INF/*.xml`	Deskriptoren für für Internet-basierte Anwendungen
MConsole-Basisverzeichnis`/css`	Stylesheet-Dateien
MConsole-Basisverzeichnis`/html`	HTML-Dateien
MConsole-Basisverzeichnis`/images`	In der Benutzeroberfläche verwendete GIF-Grafikdateien
MConsole-Basisverzeichnis`/js`	JavaSriptTM-Dateien
MConsole-Basisverzeichnis`/*.jsp`	JavaServer PagesTM-Dateien
WebConsole-Basisverzeichnis`/prereg/jesmc/*.reg`	Web Console-Dateien für die Monitoring Console

Starten der Monitoring Console

Die Monitoring Console ist eine Internetanwendung, die über einen Browser aufgerufen werden kann. Dieser stellt zu dem Host-Rechner, auf dem sie installiert ist, eine Verbindung her. Sie rufen die Monitoring Console über die Web Console auf, die automatisch auf dem gleichen Host-Rechner installiert wird. Im Folgenden wird beschrieben, wie Sie die Monitoring Console aufrufen und überwachte Komponenten anzeigen.

▼ Aufrufen der Monitoring Console

1. **Sie müssen zunächst den Web-Server für die Web Console neu starten. Führen Sie auf dem Host-Rechner bzw. in der Zone, wo Sie die Monitoring Console installiert haben, den folgenden Befehl aus:**

   ```
   /usr/sbin/smcwebserver restart
   ```

2. **Warten Sie, bis die Web Console gestartet wird. Mit dem folgenden Befehl sehen Sie, ob sie bereit ist:**

   ```
   /usr/sbin/smcwebserver status
   ```

Es kann sein, dass Sie diesen Befehl mehrmals ausführen müssen, bis die folgende Meldung angezeigt wird:

```
Sun Java(TM) Web Console wird ausgeführt.
```

3 **Rufen Sie die Web Console mit der folgenden URL auf. Verwenden Sie dazu einen Browser, der zum Host-Rechner der Monitoring Console eine Verbindung herstellen kann. Bei Installation in einer Solaris Zone ist *MC-Host* der logische Hostname, den Sie dieser Zone gegeben haben:**

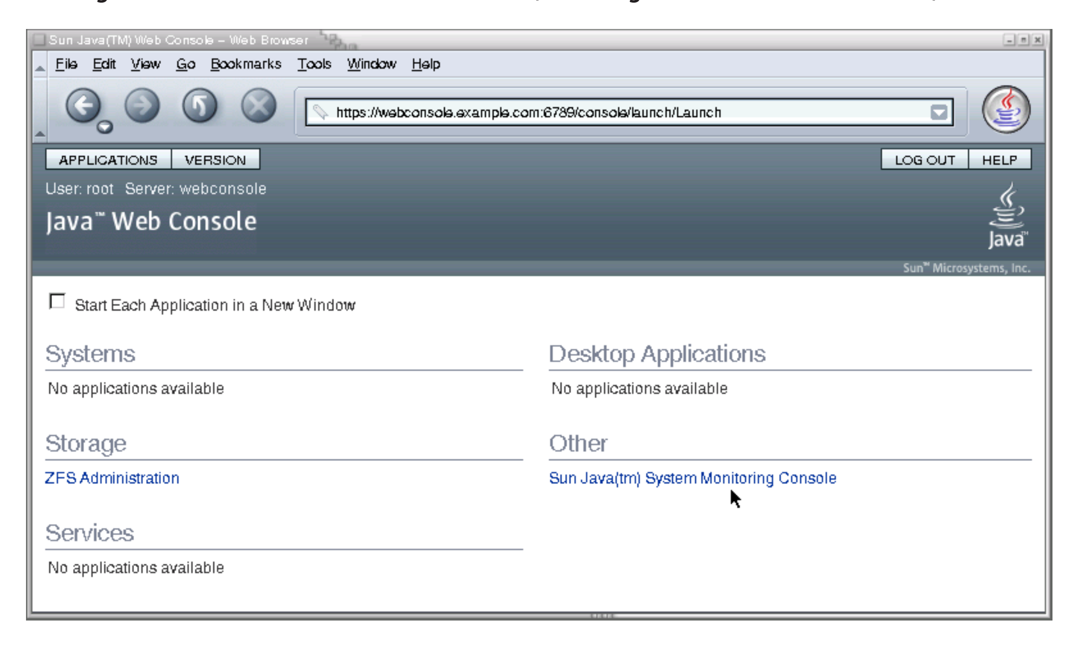
`https://`*MC-Host*`.`*Domäne*`:6789`

4 **Je nach der Konfiguration Ihres Browsers kann es sein, dass eine Meldung über ein nicht vertrauenswürdiges Zertifikat angezeigt wird. Um die Web Console aufrufen zu können, müssen Sie in diesem Fall die Vertrauenswürdigkeit des Zertifikats bestätigen.**

5 **Melden Sie sich an der Web Console mithilfe des root-Passworts für den Host-Rechner der Monitoring Console als `root` an, wenn Sie dazu aufgefordert werden.**

Nach der Anmeldung listet die Web Console alle Dienste auf, die sie anbietet.

6 **Klicken Sie zum Öffnen des Hauptfensters der Monitoring Console auf Sun Java System Monitoring Console unter der Überschrift "Weitere" (siehe folgender Bildschirmausschnitt).**



▼ Herstellen von Verbindungen zu den Knotenagenten

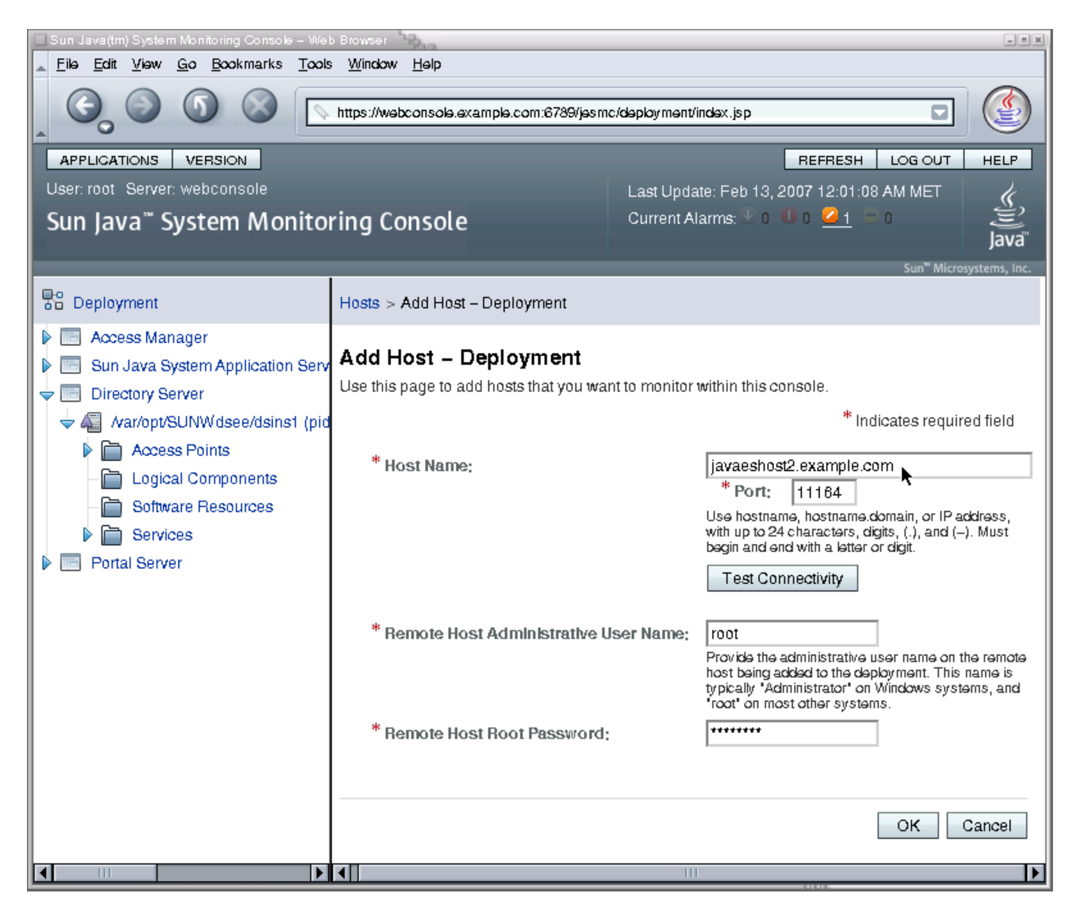
Beim allerersten Start der Monitoring Console müssem Sie angeben, wo sich die überwachten Komponenten befinden. Der Installationsort eines jeden Knotenagenten wird in der Java ES-Bereitstellung angegeben, und die Konsole zeigt automatisch alle Komponenten in jedem Knotenagenten an. Sie müssen diesen Vorgang später wiederholen, wenn Sie Ihre Bereitstellung auf den jeweiligen Host-Rechnern um neue Java ES-Komponenten erweitern.

Nach dem Hinzufügen eines Knotenagenten stellt die Monitoring Console immer dann eine Verbindung zu diesem her, wenn Sie die Konsole aufrufen. Dies geschieht solange, bis sie den betreffenden Knotenagenten explizit entfernen Falls ein vorher hinzugefügter Knotenagent nicht reagiert, sollten Sie die Anweisungen im Abschnitt „Neustart eines Knotenagenten" auf Seite 30 abarbeiten.

1. **Synchronisieren Sie Datum und Uhrzeit zwischen dem logischen Host, auf dem die Monitoring Console installiert ist, und dem Host-Rechner, auf dem der Knotenagent und die zu überwachenden Java ES-Komponents laufen. Unabhängig davon, ob die Synchronisierung automatisch oder manuell durchgeführt wird, sollte der Zeitunterschied zwischen beiden Host-Rechnern nicht mehr als 10 Minuten betragen.**

2. **Gehen Sie, falls erforderlich, durch Klicken auf den Verweis "Bereitstellung" (in der obersten Hierarchie-Ebene links in der Monitoring Console) zur Anzeige auf Bereitstellungsebene. Wählen Sie im rechten Teilfenster die Registerkarte "Hosts" und klicken Sie auf "Hinzufügen".**

3. **Geben Sie in das nun angezeigte Dialogfeld "Host hinzufügen" die erforderlichen Informationen ein (siehe folgender Bildschirmausschnitt):**



- Hostname: Geben Sie den vollständigen Hostnamen eines Knotenagenten ein, wo Sie überwachte Komponenten konfiguriert haben.
- Port: 11164, es sei denn, Sie haben die Monitoring Framework auf dem Host, wo der Knotenagent installiert ist, anderweitig konfiguriert.
- Root-Passwort für Remote-Host: Geben Sie das Root-Passwort für das System ein, auf dem der Knotenagent installiert ist.

4. **Klicken Sie auf "Konnektivität testen".**

   Wenn die Verbindungsinformationen richtig sind und der Host-Agent konfiguriert ist und ausgeführt wird, zeigt das Dialogfeld an, dass die Verbindung hergestellt ist.

5. **Klicken Sie auf "OK", um das Dialogfeld "Host hinzufügen" zu schließen. In der Hostliste wird jetzt der neue Name angezeigt. Alle vom Knotenagenten dieses Hosts überwachten Komponenten werden ebenfalls in der linken Spalte angezeigt.**

6 **Wiederholen Sie diesen Vorgang für jeden Host in Ihrer Java ES-Bereitstellung, auf dem überwachte Komponenten installiert sind.**

Nächste Schritte Sie können jetzt durch die in der linken Spalte angezeigten Komponenten gehen und ihren Betriebsstatus, die überwachten sichtbaren Attribute und eventuell ausgelöste Alarme anzeigen.

Verwenden der Monitoring Console

In diesem Abschnitt wird die Interaktion mit der Monitoring Console beschrieben.

▼ Selektives Deaktivieren und Neuaktivieren der Überwachung

Die Java ES-Überwachungsmechanismen sind so ausgelegt, dass sie die Leistungsfähigkeit von Produktionssystemen in der geringstmöglichen Weise beeinflussen. In einigen Fällen kann es jedoch sein, dass das Erfassen von Überwachungsdaten vollständig eingestellt werden muss, damit sich die Instrumentierung so gut wie gar nicht auf die Leistung auswirkt. In der Monitoring Console kann dies für jeden einzelnen Host konfiguriert werden (siehe folgende Anleitung).

1 **Gehen Sie, falls erforderlich, durch Klicken auf den Verweis "Bereitstellung" (in der obersten Hierarchie-Ebene links in der Monitoring Console) zur Anzeige auf Bereitstellungsebene. Klicken Sie dann im rechten Teilfenster auf die Registerkarte "Hosts".**



Sie führt weiterhin alle Hosts auf, die von der Monitoring Console überwachte Java ES-Komponenten enthalten.

2. **Mithilfe der Kontrollkästchen in der linken Tabellenspalte können Sie alle Hosts deaktivieren, die nicht überwacht werden sollen. Klicken Sie dann oben in der Hosttabelle auf "Deaktivieren".**

Weitere Informationen:

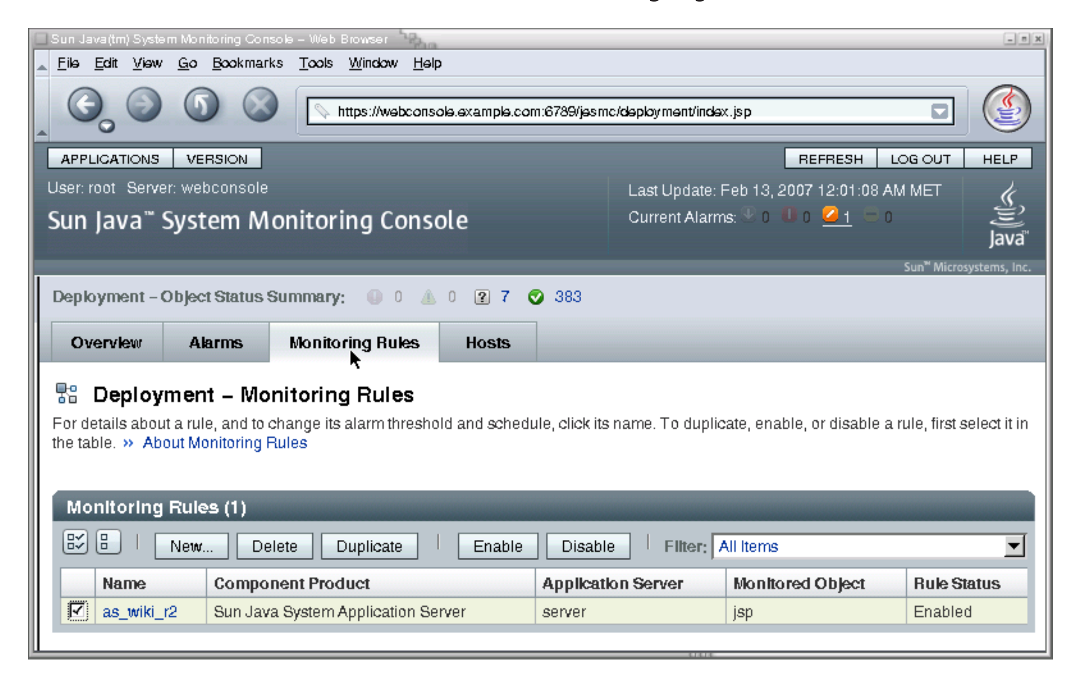
Folgen

Wenn die Überwachung für einen Host deaktiviert wird, werden auch alle in der Hierarchie für diesen Host angezeigten überwachten Objekte deaktiviert. Im deaktivierten Zustand wird die Anzeige überwachter Objekte nicht mehr aktualisiert, obwohl sie noch die zuletzt erfassten Werte enthalten kann Überwachungsregeln, die von einem deaktivierten Objekt abhängen, werden ausgesetzt. Arbeiten Sie die obige Anleitung ab, um einen vorher deaktivierten Host wieder zu aktivieren. Diesmal müssen Sie jedoch auf die Schaltfläche "Aktivieren" klicken

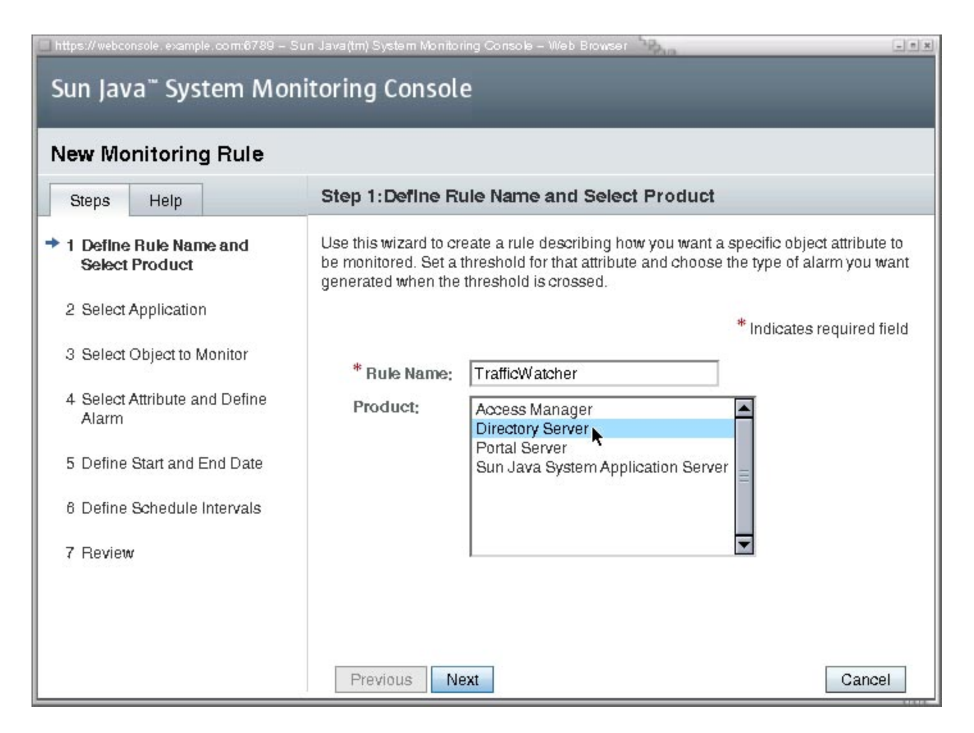
▼ Erstellen einer neuen Überwachungsregel

Eine Überwachungsregel (manchmal auch als Überwachungsaufgabe bezeichnet) ist ein Satz von Bedingungen, die vom Benutzer zum Auslösen eines Alarms definiert werden. Mit dem Überwachungsregel-Assistent der Monitoring Console können Sie die Bedingungen definieren, die überwacht werden sollen.

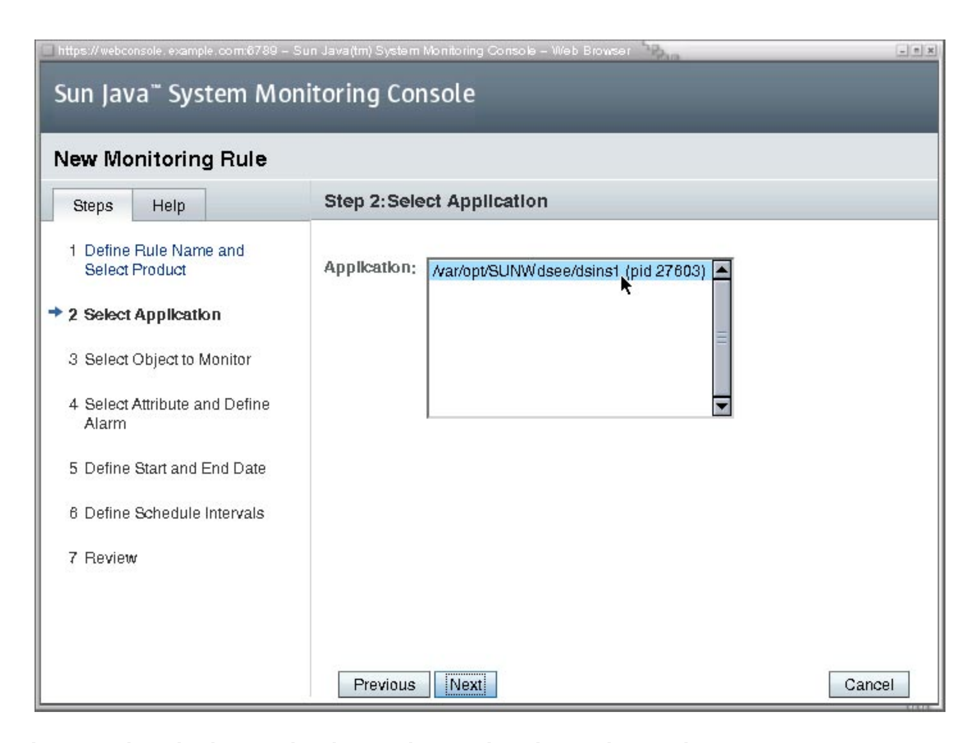
1 Gehen Sie, falls erforderlich, durch Klicken auf den Verweis "Bereitstellung" (in der obersten Hierarchie-Ebene links in der Monitoring Console) zur Anzeige auf Bereitstellungsebene. Wählen Sie im rechten Teilfenster die Registerkarte "Regeln" (siehe folgender Bildschirmausschnitt) und klicken Sie dann in der Überwachungsregeltabelle auf "Neu":



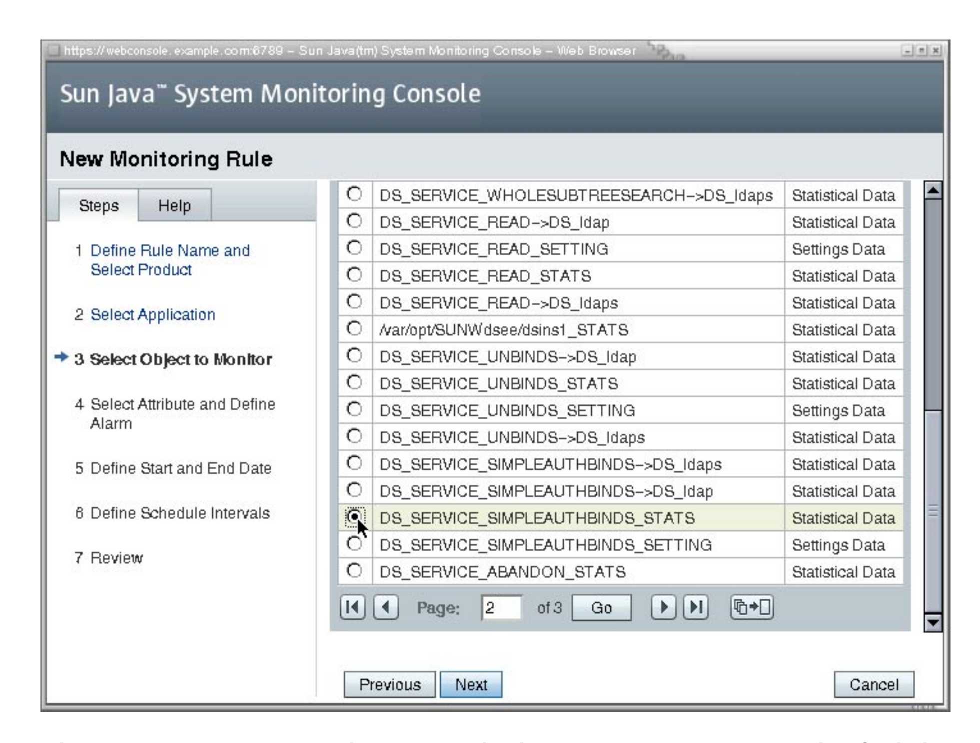
2 Geben Sie der neuen Überwachungsregel einen Namen und wählen Sie den Servertyp aus, der überwacht werden soll.



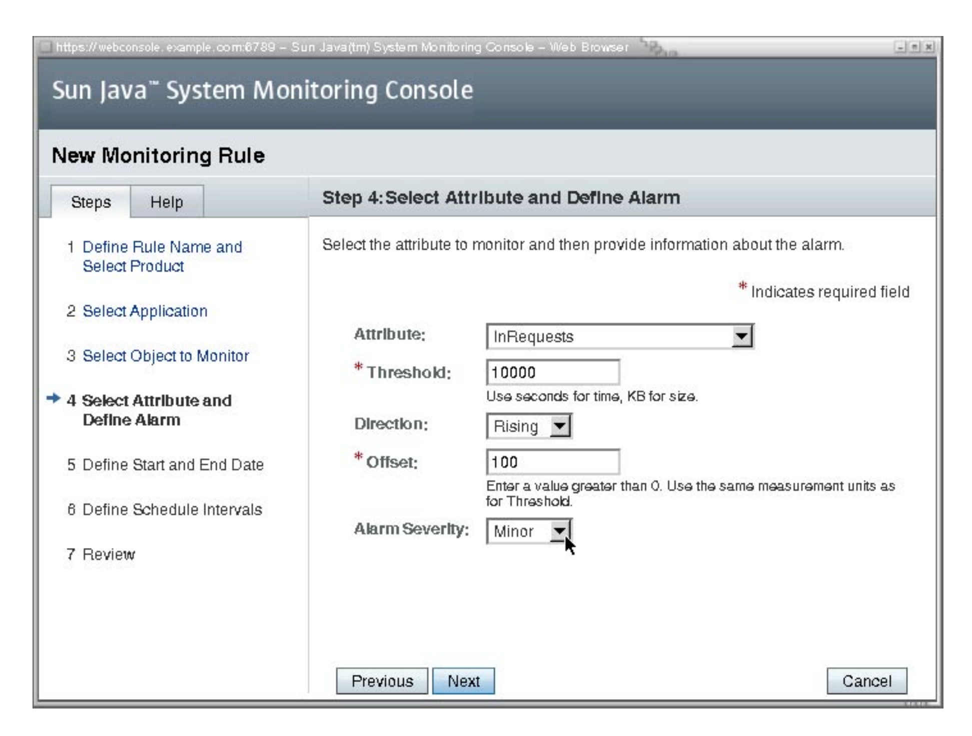
3. **Wählen Sie die Instanz der zu überwachenden Produktkomponente aus. Falls zwei Instanzen des gleichen Produkts auf unterschiedlichen Hosts installiert sind, kann es sein, dass diese Instanzen in der Tabelle mit den gleichen Namen aufgeführt sind. In diesem Fall können die Instanzen in der gleichen Reihenfolge auftreten, wie sie in der Hierarchie im linken Teilfenster angezeigt werden, dies ist jedoch nicht sicher zu ermitteln. Unter Umständen müssen Sie für beide Instanzen Überwachungsregeln erstellen, um sicherzugehen, dass die gewünschte Regel auch wirklich definiert wird.**



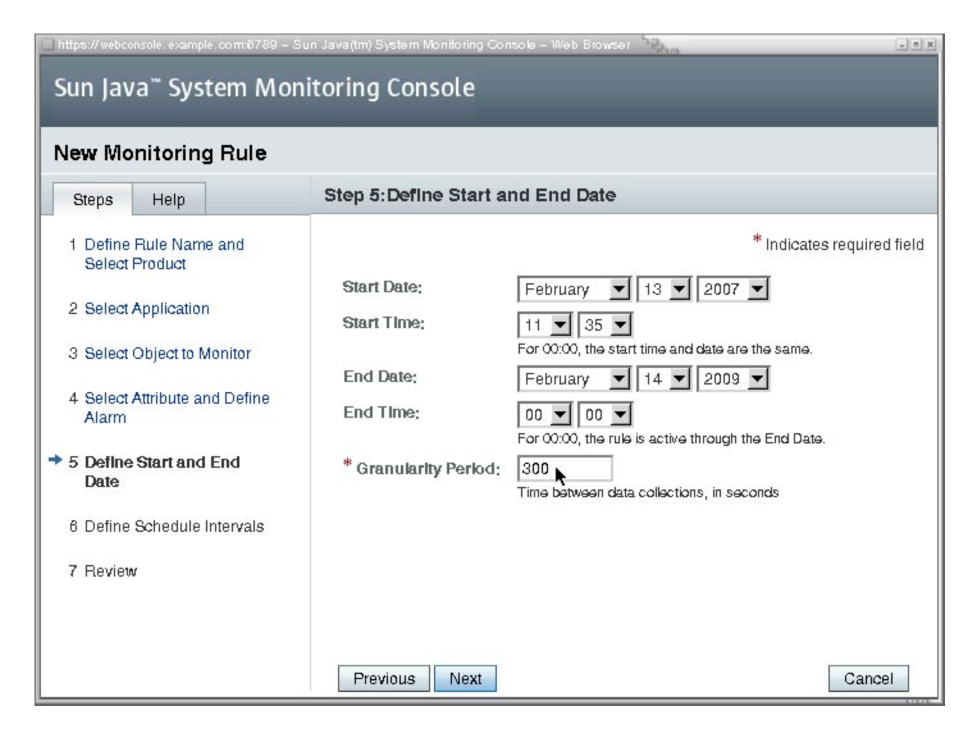
4 Wählen Sie das Objekt aus, das das zu überwachende Attribut enthält:



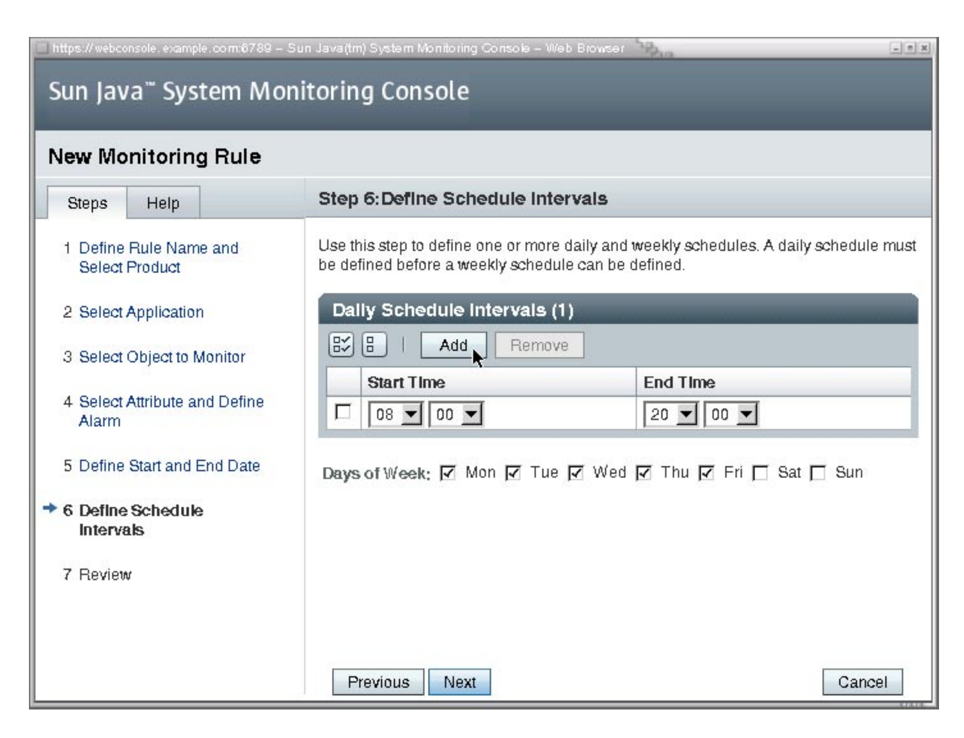
5 **Jetzt können Sie zusammen mit den Werten, die den Alarm auslösen sollen, schließlich das zu überwachende Attribut auswählen.**



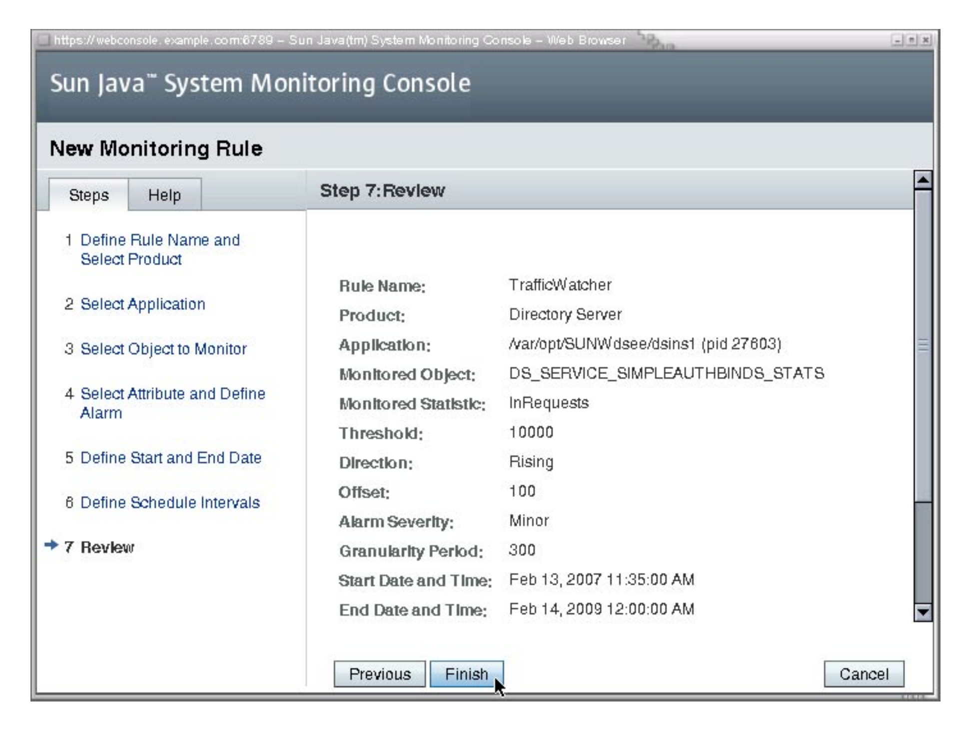
6 **Geben Sie für die Regel ein Start- und Enddatum ein. Im Gegensatz zum Plan, der die Aktivität bzw. Inaktivität einer Regel festlegt, wird durch das Start- und Enddatum definiert, wann eine Regel gelten soll. Wenn eine Startzeit in der Vergangenheit liegt (wie das beim Standardwert stets der Fall ist), beginnt die Überwachung einer zugehörigen Regel sofort.**



7 Alternativ können Sie mit den Bedienelementen auch mehrere Zeiträume festlegen, in denen die betreffende Regel gelten und eine aktive Überwachung durchführen soll. Durch Auswahl von Wochentagen können Sie auch einen wöchentlichen Plan festlegen.



8 **In diesem letzten Schritt im Regelassistent können Sie die von Ihnen gemachten Eingaben überprüfen. Klicken Sie dann auf "Fertig stellen", um die neue Regel zu erstellen.**



Nach dem Beenden des Regelassistenten wird die Registerkarte “Überwachungsregeln” wieder angezeigt, und die neue Regel erscheint in der Regeltabelle.

Fehlerbehebung an der Monitoring Console

Bekannte Probleme sind auch in den *Sun Java Enterprise System 5 Versionshinweise für UNIX* aufgeführt.

Falls Konflikte zwischen Master-Agent und Knotenagenten auftreten, sollten Sie Folgendes überprüfen:

- Bei der Verwendung von Solaris Zones sollten Sie sich vergewissern, dass die Monitoring Console in einer lokalen Sparse-Root-Zone installiert ist.
- Stellen Sie sicher, dass sich auf dem Host-Rechner bzw. der Zone keine andere Installation der überwachten Komponente befindet.
- In beiden Fällen müssen Sie die Monitoring Console und eventuelle Komponenten deinstallieren, das Problem beheben und Monitoring Console neu installieren.

Wenn Sie die Monitoring Console deinstallieren und anschließend auf dem gleichen Host neu installieren wird sie nicht initialisiert und in der Web Console nicht angezeigt. In diesem Fall

müssen Sie auf dem Host der Monitoring Console den Befehl `masetup –i` ausführen, um den Master-Agenten zu initialisieren. Arbeiten Sie dann die Anleitung im Abschnitt „Aufrufen der Monitoring Console" auf Seite 57 ab.

Überwachungsregeln können nur deaktiviert werden, wenn sie aktiv sind. Wenn Sie eine Regel deaktivieren wollen, deren Plan sie gegenwärtig deaktiviert hat, müssen Sie entweder den Plan entsprechend ändern, um sie kurzfristig zu aktivieren oder die gesamte Regel löschen.

Aufgrund von Einschränkungen auf der Windows-Plattform sind die Werte `handleCount` und `threadCount` in der Hostatistik stets 0 (null).

ANHANG A

CMM-Objektreferenz

Das Common Monitoring Model (CMM) ist eine Erweiterung des Common Information Model (CIM), das in der Programmiersprache Java implementiert ist. CIM ist in den Java-Schnittstellen der Pakete com.sun.cmm.cim.* enthalten. CMM ist darüber hinaus in den Paketen `com.sun.cmm.*` enthalten, die die CIM-Schnittstellen erweitern. Überwachte Objekte sind in den Knotenagenten als Klassen enthalten, die die CMM-Schnittstellen implementieren. In den folgenden Tabellen ist aufgeführt, welche Attribute für jede Objektklasse überwacht werden können.

Überblick über CMM-Objekte

CMM beruht auf einem begrenzten Satz von Kernschnittstellen, die definieren, welche Attribute eines überwachten Objekts öffentlich zugänglich sind. In der folgenden Liste sind die Klassen, die die zur Überwachung vorgesehenen CMM-Objekte darstellen, sowie einige ihrer Schlüsselattribute aufgeführt:

`CMM_InstalledProduct`
: Eine Java ES-Produktkomponente in ihrer Gesamtheit. Beispiel: Java ES Directory Server.

`CMM_ApplicationSystem`
: Eine installierte und konfigurierte Instanz einer Java ES-Produktkomponente. Eine solche Instanz kann entweder ausgeführt werden oder nicht. Typische Attribute eines solchen Objekts sind beispielsweise Kontaktinformationen für den Administrator, der Betriebsstatus des Systems oder Start- und Stoppzeit einer Anwendung.

`CMM_Service`
: Eine bestimmte Funktion einer Produktkomponente, beispielsweise der Authentifizierungsdienst von Java ES Directory Server. Ein typisches Attribut wäre der Betriebsstatus des Dienstes.

`CMM_SoftwareResource`	Eine Darstellung von Software-Entitäten in der Umgebung wie z. B. einem Cache, Thread-Pool usw. Ein typisches Attribut wäre die Größe eines Cache-Speichers.
`CMM_LogicalComponent`	Eine Entität, die von einem Dienst bearbeitet wird und für den Endbenutzer sichtbar ist, jedoch kein eigentliches Hardwaremodul oder eine Softwarefunktion darstellt. Bei einer logischen Komponente handelt es sich beispielsweise um einen Satz Konfigurationsparameter für eine Softwareinstanz, anstatt um die Instanz selbst.
`CMM_ServiceAccessURI`	Der Punkt, an dem ein Dienst zur Verwendung verfügbar gemacht wird. Typische Attribute sind beispielsweise eine Port-Nummer oder ein Uniform Resource Identifier (URI).
`CMM_RemoteServiceAccessPoint`	Zugriffs- und Adressinformationen für eine Netzwerkverbindung. Typische Attribute sind beispielsweise ein URI oder der Betriebsstatus einer Verbindung (offen oder geschlossen).
`CMM_Process`	Eine einzelne Instanz eines ausgeführten Programms. Typische Attribute sind beispielsweise Speicherplatz oder CPU-Auslastung.
`CMM_UnitaryComputerSystem`	Ein einzelner, von einer Java ES-Bereitstellung genutzter Host-Rechner, z. B. ein Desktop-Computer oder ein Server. Typische Attribute könnten beispielsweise die Anzahl der verfügbaren Prozessoren oder physische Speicherplatz sein.
`CMM_OperatingSystem`	Die Software bzw. Firmware, die die Hardware eines Host-Rechners nutzbar macht. Ein typisches Attribut wäre beispielsweise der verfügbare virtuelle Speicherplatz auf einem System.
`CMM_JVM`	Die von einem Java ES-Server verwendete Java Virtual Machine. Ein Beispielattribut könnte die Versionsnummer der Java Virtual Machine sein.
`CMM_DatabaseService`	Eine im Auftrag einer Datenbank ausgeführte Aufgabe, z. B. Bereitstellung eines Benutzerzugangs. Ein typisches Attribut könnte beispielsweise die maximale zulässige Anzahl an Verbindungen zur Datenbank sein.

`CMM_CommonDatabase`	Eigenschaften, die für einen bestimmten Datenbanktyp gleich sind. Ein typisches Attribut könnte beispielsweise das Datum der zuletzt gemachten Sicherungskopie sein.



ANHANG B

Objekte, die bei jeder Komponente zur Überwachung vorgesehen sind

In diesem Anhang sind die CMM-Objekte aufgeführt, die in jeder Produktkomponente, die Überwachung unterstützt, zur Überwachung instrumentiert sind. Wenn nur ein Teil von Objektattributen instrumentiert ist, sind diese Attribute ebenfalls aufgeführt.

Instrumentierung des Common Agent Container

Noch nicht dokumentiert.

Instrumentierung von Access Manager

Noch nicht dokumentiert.

Instrumentierung von Anwendungsserver

Noch nicht dokumentiert.

Instrumentierung von Calendar Server

Noch nicht dokumentiert.

Instrumentierung von Directory Server

Noch nicht dokumentiert.

Instrumentierung von Instant Messaging

Noch nicht dokumentiert.

Instrumentierung von Messaging Server

Noch nicht dokumentiert.

Instrumentierung von Portal Server

Noch nicht dokumentiert.

Instrumentierung von Web Server

Noch nicht dokumentiert.

Index